UNSTRUCTURED ROAD RECOGNITION AND FOLLOWING FOR MOBILE ROBOTS VIA IMAGE PROCESSING USING ANNs

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

RASİM AŞKIN DİLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

JUNE 2010

Approval of the thesis:

UNSTRUCTURED ROAD RECOGNITION AND FOLLOWING FOR MOBILE ROBOTS VIA IMAGE PROCESSING USING ANNs

submitted by **RASİM AŞKIN DİLAN** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Suha Oral _____
Head of Department, **Mechanical Engineering**

Assist. Prof. Dr. A. Buğra Koku _____
Supervisor, **Mechanical Engineering Dept., METU**

Assist. Prof. Dr. E. İlhan Konukseven _____
Co-supervisor, **Mechanical Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Reşit Soylu _____
Mechanical Engineering Dept., METU

Assist. Prof. Dr. A. Buğra Koku _____
Mechanical Engineering Dept., METU

Assist. Prof. Dr. E. İlhan Konukseven _____
Mechanical Engineering Dept., METU

Assist. Prof. Dr. Melik Dölen _____
Mechanical Engineering Dept., METU

Assoc. Prof. Dr. Veysel Gazi _____
Electrical and Electronics Engineering Dept., TOBB ETU

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Rasim Aşkın DİLAN

Signature :

ABSTRACT

UNSTRUCTURED ROAD RECOGNITION AND FOLLOWING FOR MOBILE ROBOTS VIA IMAGE PROCESSING USING ANNs

Dilan, Rasim Aşkın
M.Sc., Department of Mechanical Engineering
Supervisor : Assist. Prof. Dr. A. Buğra Koku
Co-Supervisor : Assist. Prof. Dr. E. İlhan Konukseven

June 2010, 169 pages

For an autonomous outdoor mobile robot ability to detect roads existing around is a vital capability. Unstructured roads are among the toughest challenges for a mobile robot both in terms of detection and navigation. Even though mobile robots use various sensors to interact with their environment, being a comparatively low-cost and rich source of information, potential of cameras should be fully utilized. This research aims to systematically investigate the potential use of streaming camera images in detecting unstructured roads. The investigation focused on the use of methods employing Artificial Neural Networks (ANNs). An exhaustive test process is followed where different kernel sizes and feature vectors are varied systematically where trainings are carried out via back-propagation in a feed-forward ANN. The thesis also claims a contribution in the creation of test data where truth images are created almost in real-time by making use of the dexterity of human hands. Various road profiles

ranging from human-made unstructured roads to trails are investigated. Output of ANNs indicating road regions is justified against the vanishing point computed in the scene and a heading vector is computed that is to keep the robot on the road. As a result, it is shown that, even though a robot cannot fully rely on camera images for heading computation as proposed, use of image based heading computation can provide a useful assistance to other sensors present on a mobile robot.

Keywords: Unstructured Road Recognition, Mobile Robotics, Artificial Neural Network Training, Pattern Recognition

ÖZ

HAREKETLİ ROBOTLAR İÇİN YAPAY SİNİR AĞLARI KULLANILARAK GÖRÜNTÜ İŞLEME İLE DÜZENSİZ YOL TANIMA VE TAKİBİ

Dilan, Rasim Aşkın
Yüksek Lisans, Makine Mühendisliği Bölümü
Tez yöneticisi : Yrd. Doç. Dr. A. Buğra Koku
Ortak tez yöneticisi: Yrd. Doç. Dr. E. İlhan Konukseven

Haziran 2010, 169 sayfa

Otonom bir hareketli arazi robotu için çevresinde bulunan yolları algılayabilmesi önemli bir özelliktir. Düzensiz yollar, hareketli robotlar iç yol tanıma ve yol takibi açısından en önemli zorluklardan biridir. Hareketli robotlar çevreleri ile etkileşimlerini sağlayan çeşitli duyaçlar kullanmalarına rağmen, diğer duyaçlarla karşılaştırıldığı zaman daha az maliyeti olan ve zengin bir bilgi edinimi sağlayan kameraların potansiyelleri tamamen kullanılmalıdır. Bu araştırma düzensiz yolların tanınması için akan kamera görüntülerinin potansiyel kullanımını soruşturmaktadır. Araştırma, Yapay Sinir Ağları'nın kullanıldığı yöntemlerin kullanımı üzerine yoğunlaşmıştır. Farklı nüve büyüklüklerinin ve öznitelik vektörlerinin sistematik olarak değiştirilmesi ile oluşturulan vektörler ile çeşitli İleri Beslemeli Yapay Sinir Ağları'nın geriye yayılım yöntemi ile eğitildiği kapsamlı bir test süreci izlenmiştir. Tezde aynı zamanda insan ellerinin çevikliğini kullanarak neredeyse gerçek zamanlı bir biçimde test verilerinin yaratılması konusunda bir katkı sağlanmıştır. İnsanlar tarafından yapılmış düzensiz yollardan patikalara kadar çeşitli yol

profilleri incelenmiştir. Yapay Sinir Ağları'nın çıktı olarak verdiği hesaplanmış yol alanı görüntüdeki kaçış noktası bulunarak doğrulanmakta, ve bu veriler ışığında robotu üzerinde bulunduğu yol içinde tutmaya yarayacak bir yön vektörü hesaplanmıştır. Sonuç olarak, robot sadece kamera görüntüleri ile hesaplanmış referans yön vektörüne itibar etmese de, sadece görüntüler kullanılarak hesaplanan referans yön vektörünün hareketli robot üzerinde bulunan diğer duyaçlara yararlı bir şekilde destek verebileceği gösterilmiştir.

Anahtar kelimeler: Düzensiz Yol Tanıma, Hareketli Robotlar, Yapay Sinir Ağı Eğitimi, Örüntü tanıma

To My Dear Mother and Father

To My Sister

And

To My Love Tuğçe

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere appreciation to my supervisors Asst. Prof. Dr. A. Buğra Koku, Asst. Prof. Dr. E. İlhan Konukseven and Asst. Prof. Dr. Melik Dölen for their invaluable guidance, advice, and criticism and support that made this study possible.

I would like to express my thanks to my SPARC LAB colleagues Barış Ragıp Mutlu, Serdar Üşenmez and Ulaş Yaman for their friendship and technical support throughout the thesis period. I would also like to thank my Automatic Control Lab colleague Hakan Çalışkan for his support during all sleepless nights we worked in the lab. I want to thank to my friend Hatice Sinem Şaş for encouraging me during the desperate times of this study.

Also, I would like to thank my family, Tunçay and Ergin Dilan, and my sister, Hasibe Betül Dilan, for their understanding and help.

I would like to thank my dear love, Tuğçe Yalçın, for changing my life and being there for me all the time. It is a great feeling to know that somebody cares about you and will be by your side in every situation. It is good to know that there is somebody to share your innermost thoughts, somebody to listen to you even if she knows that you are wrong. Moreover I want to thank Rose, my sweetest, for being such pretty and giving joy to my life.

Last, but not least, I thank the Scientific and Technological Research Council of Turkey for their financial support with grant code BIDEB2228 in the research that made it possible for me to conduct this thesis work.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF TABLES

TABLES

LIST OF SYMBOLS

SYMBOLS	
N	Total number of pixels in an image
P_i	Probability of i^{th} level pixel
μ_T	Grand average of an image's gray levels
σ_{BC}	Between-class variance
S	Separation factor
ν_T	Total variance
$\underline{H}$	Homography matrix
H_{ij}	Element at i^{th} row and j^{th} column of homography matrix
(u_i, v_i)	Coordinates in the plane before homography projection
(u_i', v_i')	Coordinates in the plane after homography projection
λ	Wavelength
g_{odd}	Gabor filter (odd phase)
g_{even}	Gabor filter (even phase)
k	Kernel size
$I_{complex}$	Complex response of Gabor filter
IP	Image pixel
θ	Dominant orientation of image pixel
VC	Vanishing point candidate
FVI	Feature vector depending on independent histograms
FVJ	Feature vector depending on joint histograms
FVA	Feature vector depending all kernel information

$\underline{TM}$	Truth matrix
κ	Kappa coefficient
P_{RR}	Probability dist. of road points in both images
P_{RN}	Probability dist. of road points in image#1 and non-road points in image#2
P_{NR}	Probability dist. of non-road points in image#1 and road points in image#2
P_{NN}	Probability distribution of non-road points in both images
$P_{\_N}$	Probability distribution of all non-road points in image#2
$P_{\_R}$	Probability distribution of all road points in image#2
$P_{R\_}$	Probability distribution of all road points in image#1
$P_{N\_}$	Probability distribution of all non-road points in image#1
P_O	Observed level of agreement ratio
P_E	Expected value of agreement ratio
CDP	Correctly detected point
IDP	Incorrectly detected point

CHAPTER 1

INTRODUCTION

1.1 On Mobile Robotics

Mobile robotics has been an active research area for the past decades, since it involves several academic branches and has a depth in all of these branches. Control engineering, artificial intelligence and cognitive science, pattern recognition and digital image processing fields are some of the examples for those branches. Due to immature solutions to the problems of mobile robotics and increasing demand of autonomous mobile applications, the interest on this area has become intense.

Mobile robots are robots having the capability of autonomous movement in the settings where they are placed. While moving in the environment they are placed, they are expected to accomplish several tasks or to collect information about that environment. The mobile capabilities of these robots reveal several application areas for these robots. Such robots can be utilized for transportation, surveillance, guidance, exploration, inspection or several other tasks.

There are lots of mobile robot application areas in real life. For instance, there are hospital robots which are able to make predictable, anywhere-to-anywhere deliveries avoiding people and other obstacles in hospitals.

Several projects need remote access to distant facilities. Mobile platforms collecting sensor data and images for tracking and real-time observation of these sites are also available. Outdoor mobile robots offer multiple solutions for commercial applications. Especially, for environments that are inaccessible, dangerous or hostile to the livings, mobile robots are utilized. For example, a planetary robot, which explores the surface of Mars, is a type of mobile robot. Moreover, mine sweepers operating autonomously under dangerous environment are a type of mobile robots.

Beside the variety of application areas of mobile robots, autonomous mobile robots offer excellent means of testing hypothesis about intelligent behavior, perception and cognition, and become popular in artificial intelligence and cognitive science fields.

In most of the applications requiring mobile robots, localization of the robot is the most crucial problem. Mobile robot should have the knowledge of its local or global position by some means for navigation. The basic property of the mobile robots, which gives the autonomous capabilities to the robot, is its localization capability. Depending on the application, localization can be provided using a laser sensor, a camera, or any other means of sensors. Whatever the utilized sensor is, the accurate localization is the key point for mobile robot autonomy.

1.2 On Road Recognition and Following

Road recognition, detection and following problem for mobile robots has been an active research area for the past several decades. Road detection for

mobile robots is required for the environments which are dangerous for human-beings. Moreover, it can be used for assisting humans while driving or operating a vehicle.

Road detection is an important requirement for autonomous navigation even in the presence of assisting technologies such as GPS. Road recognition can be performed using sensors such as; laser sensors, omni-vision cameras, etc. and several algorithms and applications are developed in the literature offering satisfactory solutions. However, most of the satisfactory solutions cannot be applied to all types of roads a mobile robot has to deal with during autonomous navigation. Roads can be classified into two groups regarding to the setting as; structured and unstructured roads.

Research on road detection for structured roads (i.e. asphalt roads) has produced well-working solutions. Satisfactory unstructured road detection algorithms using the several sensors other than vision sensors are available in literature. However, unstructured road detection through the sole use of vision sensors is still an open research area. A detailed literature survey on road recognition and following algorithms are given in CHAPTER 2.

1.3 Scope of the Thesis

In this thesis, the primary objective is to specify the road region in an unstructured road image captured by a simple camera. The only input into the algorithm/system is the image extracted from a video stream. Other

than this visual input, no other means of sensors are used for manipulating the data or the proposed road recognition algorithm.

The study aims to systematically explore the limits of maximum amount of useful information out of a simple VGA image, tries to detect road regions using only this information and creates a recommended heading value to aid the robot in following the road lying in front of it. In other words, this work tries to evaluate the performance of algorithms enforcing the boundaries of all information extracted from images.

Moreover, using texture properties of image, vanishing points, which represent the point at where the road disappears in the image, are extracted as well. Using the road boundaries and the extracted vanishing points, a reference robot heading, is computed.

All proposed algorithms are simulated on a software package, will not be implemented on a real mobile robot so real-time computation and algorithm response will not be sought.

For road recognition, ANNs are utilized. Among all of the classification schemes there are several reasons why neural network training is selected. Unstructured roads have no regular properties that can be easily classified. Therefore, neural networks provide the flexibility of classifying such complex features. Since neural networks does not require a specified set of rules to be followed for guaranteeing a solution, it would allow a flexible solution space for classification of unstructured road features.

1.4 Outline of the Thesis

The outline of the dissertation presented is given below:

In Chapter 2, a detailed survey on road recognition and following is given and solution techniques proposed so far in literature for both structured and unstructured roads are presented.

In Chapter 3, proposed data generation algorithm, which provides training and test data generation for Artificial Neural Networks (ANNs) are described and the performance of the proposed algorithm is discussed.

Chapter 4 starts with a brief introduction about ANNs. Then, an algorithm extracting the vanishing point in the road images are defined, the performance evaluation of this algorithm is presented and best configuration of the algorithm is selected. After that, unstructured road detection algorithm solely relying on color information is described. Detailed results on performance of the road detection algorithm are presented and a thorough discussion is given to assess these results.

In Chapter 5 of the thesis, a routine, which computes a reference robot heading, utilizing the two methods defined previously is presented. This routine determines the reference heading for mobile robot. Related results and performance evaluation are also given in this chapter.

In the final chapter, conclusion on this study is presented and suggestions on future work are given.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

In this chapter, a survey on road recognition, detection and following problem for mobile robots and vehicles is presented. The algorithms using only vision sensor or vision sensor as main sensor are presented in this chapter. Road detection algorithms are classified according to the nature of the road. There are two settings present in road recognition and following problem for mobile robotics, namely, structured and unstructured roads. This chapter is organized under these two main topics. Following section presents the literature survey on structured road detection followed by a survey on unstructured road detection. Considering the fact that the proposed approach makes use of ANNs, generation of ground truth data for training purposes is obvious, yet their acquisition is not trivial. This chapter is concluded with a brief introduction on data generation methods for ANN training purposes.

2.2 Structured Road Detection

In this section, recognition algorithms available in literature, which are proposed for the detection of structured roads using vision sensors as their main data collection tool, are presented. Structured roads are typically type of roads having a certain structure, such as highways (Figure 2-1). These

roads have clear road markings such as lane and boundary lines. They have regular and knowable shapes. There are nearly no irregularities and unpredictable properties of roads in this category. Due to the markings they possess and their predictable shapes, road detection in this setting can be replaced by detecting the markers on the road. In literature, there are lots of successful solutions to this problem utilizing vision sensors only. The research related to structured road detection has come up to saturation with the successful algorithms offering satisfactory detection performances. Therefore, research interest in road detection problem has shifted to unstructured road setting in last decades.

Figure 2-1: Structured roads (a – Suburban roads, b – Highway)

Structured recognition algorithms can be grouped under three main headings. These are model based methods, feature based methods and machine learning based methods. In the following three subsections, structured road recognition algorithms utilizing mainly vision sensors for

detection are examined in the specified categories. The methods that don't seem to belong to either of these groups are examined in other methods subsection.

2.2.1 Model Based Methods

In model based algorithms, firstly a road model is proposed. This model can be constructed in several ways. The methods for construction of this model are given throughout this subsection. After the construction of the model, algorithm tries to match the road boundaries in the input image with the constructed model. Road regions can be extracted with high accuracy via model-based methods. However, for a road with relatively complex structure, the construction of a road model, which is required by methods in this group, gets complicated.

Studies on structured road recognition using various road models started in early 1990s. Crisman et al. [1] set up several road and intersection models, which are statistical and parametric models respectively, and try to match these models with road-surface likelihood image in order to determine the most likely road model. Parametric model is a polynomial equation which models the boundaries of the road as a curve. On the other hand, statistical models are mixed Gaussian color models and are set up both for roads and non-roads. RGB channels are used for the construction and for models mean values and covariance of the channels are calculated from a sample set acquired via camera. Similarly, Behringer [2] constructed only road curvature models by approximating road boundaries with polynomial functions. Then edge images are obtained from the images acquired from

the camera and by using this edge information the proposed curvature model is converged to the road shape.

In some studies, road lane markers are modeled as several quadratic curves and coefficients of these curves are updated using a Kalman Filter-Based Update scheme to extract the road or lane boundaries. Before the update procedure, edge images of the roads are extracted using several basic edge detection methods such as Canny, Prewit edge detector and Hough transform and morphological operations are held in order to improve the edge images [3-5]. Similarly, Chen et al. [6] constructed road (RMM) and line models (LMM) and by extracting lines from images through Hough transform updates these models to keep the autonomous vehicle on the road. A similar approach is adopted by Jung et al. [7] who constructed linear road model for near vision and quadratic road model for far vision cases. In a different study, parabolas are used as a road model and the parameters of these model is calculated through randomized Hough transform and a genetic algorithm [8]. There are also several approaches that fit spline curve models to lane edges extracted by utilizing Hough transform and other edge detection algorithms. [9-11].

Similarly, Aly [10] assumed that road is composed of several splines and applied RANSAC spline fitting algorithm to the road lane markers to find out the borders. However, before the spline fitting phase, road image was transformed to top view image by using inverse perspective mapping (IPM).

In a study carried out by Aufrére et al [12-13], road edges in the image is tried to be detected using a statistical model of the lane sides. Image is divided into several strips and a statistical model, which is composed of the edge coordinate and a corresponding confidence interval, is available for each strip (Figure 2-2). By using the edge information in upcoming images, the statistical model converges to the lane edges by reducing the confidence interval. Similarly, Chapius et al [14], achieved detecting road areas by recursive updating of a statistical lane model obtained by an off-line training phase.

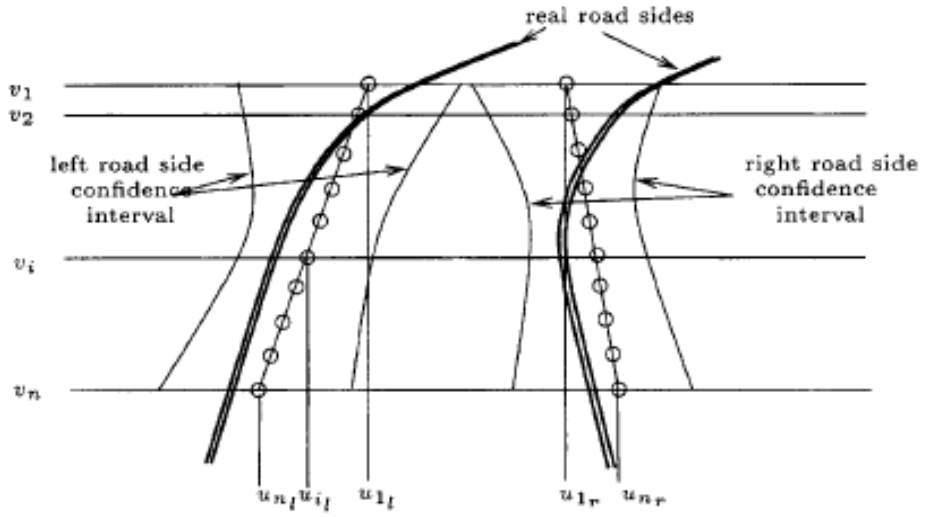

Figure 2-2: Statistical road line models and real road sides [12]

In a study held in 2004, a statistical road model containing estimated positions of lane lines, curvature and road color information is constructed. Using color information in the image, parameters of the model is attempted to be calculated [15]. Similarly, in a road following method proposed, Lin

and Zhou [16] constructed a color statistic road / non-road model. Following that, partitions of homogeneous regions were constituted up from the road image using Mean-shift algorithm. After obtaining regions, a binary image was extracted using Graph Cuts algorithm in order to update model. The model is revised by utilizing statistical color information of the model.

In a study held by Liu et al [17], vanishing points were extracted and tracked for the estimation of lane boundaries. Vanishing points are the points where the road boundaries are intersected on the horizon. Using Sobel operator, the edge image was obtained and transformed to the Hough Space. Using information of image in Hough Space, vanishing point was calculated. Then a statistical Gaussian model was constructed based on the previous vanishing point for predicting present one. Then tracking the consecutive locations of the vanishing point, lane boundaries were tried to be found.

In a different research, lane markers are extracted from the gray-scale image and then a lane model consisted of several connected lane segments with described by parameters of width, length and position is tried to fit on these extracted lane markers to obtain the boundary of the lanes in the road [18].

To summarize, every model based road extraction algorithm includes a road model and by using the information extracted from the image, it tries to have these models to converge to the actual road boundaries.

2.2.2 Feature Based Methods

In feature based algorithms, one first extracts several features related to color and edge information of the road from the color images and then can obtain the road regions by clustering, classifying these features or by region growing. The main advantage of this type of algorithms is that, they are not susceptible to the shapes of the roads and only a little priori information is required. However, these algorithms are sensitive to noises like shadows and water areas or obstacles like vehicles and pedestrians.

Studies on road recognition with classification using features started in early 1990s. Crisman et al [19] obtains a five-dimensional feature image from each RGB image, where each pixel vector x contains red, green, blue value as well as row and column id's. After obtaining the feature image, Crisman used ISODATA clustering algorithm, which is an unsupervised clustering algorithm, to mark image with non-road/road regions and form a class image. Similarly, Cheng et al [20] constructed color features from the images and then tried to cluster these features into regions using mean shift algorithm, which is also another unsupervised clustering algorithm. Likewise, in a study held by Apostolof and Zelinsky [21], color and state based features are generated and using distillation algorithm road center and its width right ahead of the vehicle is tried to be estimated.

Kreucher and Lakshmanan [22] created features using information related to the frequency domain properties of the road images using Direct Cosine Transform (DCT). Then these frequency domain features were combined

with a deformable statistical template prior for the detection of the interested lane markers.

In a study carried out by Okutomi and Noguchi [23], texture features, which are basically related to the painted areas in asphalt roads, are extracted. All images are obtained through a stereo camera system. Road region is assumed to be a plane and in order to define the transformation between stereo images and road plane, corresponding homography matrix is computed utilizing stereo image features. Then one of the images are transformed into the other plane and by pairing both images, the road region is obtained.

In another study, image data was projected to an illuminant-invariant axis and features are constructed in this space initially. A classification upon these features was then performed using a statistical region growing technique with a histogram-based classifier in order to extract road and non-road regions [24].

In a road recognition research held by Yim and Oh [25], three features, two of which are related to the starting point and direction of the lane boundary and one which is related to the gray-level intensity, were extracted using preliminary image processing methods. Then, using these features a best lane vector was chosen among the available lane vector models, which minimizes the constructed cost function.

Chern [26] chose average RGB values as the features and by using the region growing algorithm called Fischer's criterion, in order to cluster

similar small regions within the image into larger homogeneous regions. After region growing, based on the information of features that the regions, each of region is labeled and following that the borders of the regions are extracted for obtaining road borders.

To conclude for this subsection, every feature based road extraction algorithm uses features related to the color, texture or illumination properties which are extracted from the color images. After obtaining the features, several classification algorithms are used to cluster the road and non-road regions in the images.

2.2.3 Machine Learning Based Methods

Machine learning based algorithms utilizes machine learning methods to classify the images into road and non-road (background) pixels. The most common technique is to train a neural network by using features extracted from a color image as input and corresponding image which is labeled as road / non-road as output to this network.

Structured road detection using through machine learning was applied first in Carnegie Mellon University to "*Autonomous Land Vehicle in a Neural Net* (ALVINN)" system, which is an approach depends on ANNs [27-29]. Using a set of distinctive images, the neural network is trained and at the end of training phase, the robot using proposed algorithm follows the road effectively. The training inputs are stored image/steering direction pairs, which were prepared by recording the commands of a human driver during a driving scenario. After the generation of this data set, inputs were

fed to a four-layered neural network for training. However, the trained network was applicable to very limited number of roads [28].

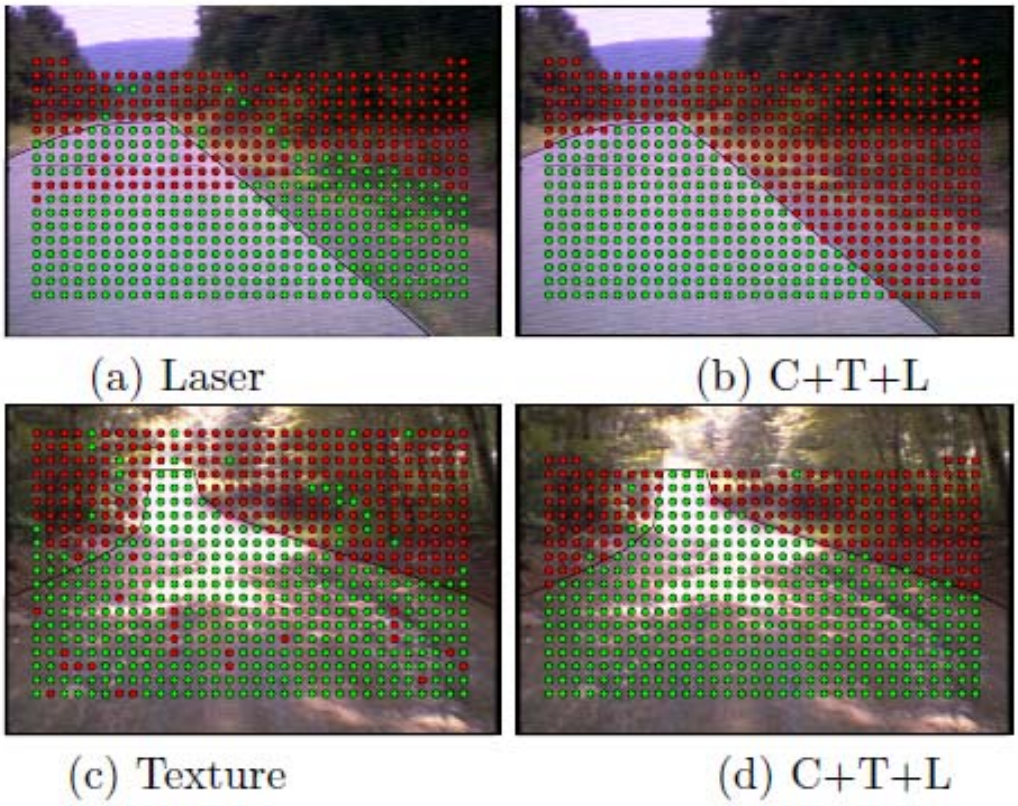

Figure 2-3: Road detection results of method using Neural Network Training. (a) structured road detection (SRD) results using only laser data (b) SRD results using all features (c) unstructured road detection (URD) results using only texture data (d) SRD results using all features (C-Color feature, T-Texture feature, L-Laser feature)

Rasmussen [30] extracted color and texture features from images captured from the on-board camera. All features were extracted for 15x15 regions in

the images. Color features are composed of joint histograms of these regions. Texture features are extracted again from these regions using Gabor filters. Other than vision-based features, features were constructed using the laser data. All these features were fed into a two layer neural network for training. The results obtained through the study are depicted in Figure 2-3 for different training schemes. The algorithm was speculated to work both on structured and unstructured roads even in ill-conditions.

In a similar study, Conrad and Foedisch [31] trained a one-layer neural network with 20 hidden neurons using the features extracted from using the independent color histogram information of 31x31 pixel sized regions of images recorded by an onboard camera. Moreover image pixel locations of 31x31 pixel sized regions were also included to feature vector. It was seen that results using features including pixel locations was superior to the one without pixel locations.

In a study carried by Oh et al [32], a control architecture based on reinforcement learning was proposed. A lane edge model was constructed and vehicle was controlled to remain between these lane edges. Lane edge models are updated using image processing algorithms while the control system was trained using the lane information extracted. After the training phase, using the image input from the on-board camera, control parameters were updated and the vehicle was controlled to stay on the road.

As a final point, in every machine learning method, features are generated from images and same images are labeled as road and non-road. Then feature vectors, as inputs, and labeled road vectors, as outputs, are supplied

into the ANN for training. After the training phase, for road detection, features extracted from new upcoming images are fed into network and the images are classified as road or non-road according to results obtained from the output of the ANN.

2.2.4 Other Methods

In this subsection, algorithms, which do not fall into the previous algorithm groups, are presented. Neither of these algorithms uses features, models or neural network training for extracting structured road regions out of the road images.

In a study carried out by Li et al [33], road recognition was achieved by fuzzy reasoning. Road image extracted from camera was compressed down to an image with a resolution of 128 x 128 pixels. Then the pixels were grouped either into a uniform region of into a border region. By this way edges of the road can be obtained with the help of fuzzy logic inference, which were defined by triangular functions such as Min-Max interference and centroid defuzzification. After the edges were extracted, the road region was specified.

Ma et al [34] used optical sensors with sensors performing W-band radar imaging in order to collect road information. Then, by fusing information from both of these sensors, using a joint MAP estimator with a Bayesian framework, road and lane boundaries were obtained.

A road recognition algorithm developed by Wang et al [35], utilized OTSU algorithm, which is a threshold segmentation algorithm, and Canny Edge Detection algorithm for road detection. Firstly, OTSU algorithm was used to extract road regions vaguely from the background. After that canny edge detection was applied to obtain the edge image. Following that the edge image was filtered using the vague road region obtained previously to finalize the road detection. In a study carried out by Wang et al [36] again, the performance of the previous study was boosted using Monte-Carlo method. By using Monte Carlo method, road boundaries belonging to consecutive image frames could be associated in order to overcome artifacts caused by illumination variations and surface irregularities.

In a research held by Kim et al [37], road images captured by an onboard camera was treated as space variant distance weights and by using a Cellular Neural Network based dynamic programming, and optimal path, which would turn out to be the boundaries of the road, was calculated. In a similar study, Lin et al [38-39] also used dynamic programming for road boundary detection. After using a Randomized Hough transform to extract the edges from gray level road image, dynamic programming was used as a spatial filter to remove the unrelated edge candidates as road boundaries. After the removal of unrelated edges, the remaining ones constituted the true road boundary.

Nourine et al [40] carried out a research for painted roads with low curvature. In this study, lane markings are extracted Radon transform, and by using the position information of the lane markings from previous frames, algorithms do not need to process whole image. Fusing the

information of previous frames and the detection result of Radon transform in the most recent frame, algorithm is able to track the lane boundaries of the road.

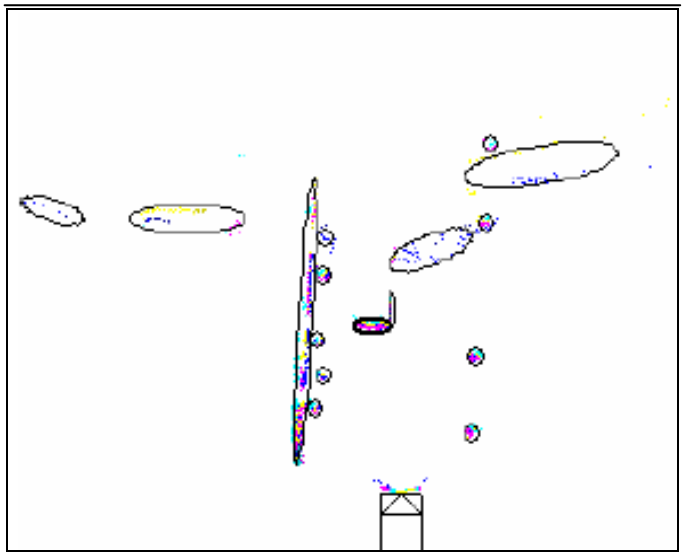

Figure 2-4: Laser Rangefinder data of a sample road [41]

In a study, Xu [41] stated that vision based algorithms were not robust in ill-conditioned scenes. Therefore, he proposed a road detection algorithm with the help of laser rangefinder data instead of vision-based algorithms. Laser rangefinder sensor is placed under the vehicle and proposed algorithm extracts the road-side grasses, bushes and trees using laser rangefinder data in order to locate the road regions (Figure 2-4). However, for several roads which do not have such obstacles, the algorithm is ineffective.

Chewputtanagul and Jackson [42] presented a road recognition system utilizing Geographical Information System (a.k.a GIS) and Global Positioning System (a.k.a GPS) data together. Recognition system finds out the vanishing point of the road and the elevation angle between the current vehicle and the vanishing point that was found out. After that, the system performs a map matching procedure to discover vehicle location.

To summarize, the algorithms belong to the other methods subsection do not use one of the methods specified in the previous three subsections directly but they utilizes some other uncommon, more authentic methods for road detection.

2.3 Unstructured Road Detection

In this section, recognition algorithms available in literature, which are proposed for detection of unstructured roads, are presented. Unstructured roads, as seen in Figure 2-5, are roads with no significant artificial markings that could be perceived and tracked for navigation. Compared with structured roads, unstructured roads are more difficult to detect. For example, the road edge border may be unclear and have low intensity contrast; the overall road shape may be arbitrary; and the road surface can be degraded in appearance [43].

There is no general dependable way of detecting roads in an off-road setting. A variety of methods is offered in the literature for detecting unstructured roads. Unstructured recognition methods can be grouped under three main groups. These are model based methods, feature based

methods and machine learning based methods. In the following three subsections, unstructured road recognition methods are examined in the specified categories. The methods that don't belong to these groups are examined in the other methods subsection.

Figure 2-5: Unstructured roads (METU Campus)

2.3.1 Model Based Methods

In model based algorithms, a road model should be constructed, which is possible in several ways which are given throughout this subsection. Following model construction, proposed algorithm tries to update the model according to the input image until it converges to the road region. Road regions can be obtained using this kind of methods at high accuracy. However, for a road with relatively complex structure, the construction of a road model, which is required by methods in this group, gets complicated.

In a study carried out by Ekinci et al [44], road detection algorithm was developed around a constructed road model. Road boundaries were modeled as two parallel curves defined by a second order polynomial. Than road was tried to be segmented using the gray level and texture feature obtained via Roberts operator. After the segmentation the road model was updated to obtain exact road boundaries. By checking the discontinuity of the road boundaries, this algorithm was also said to be able to detect the junctions along the road.

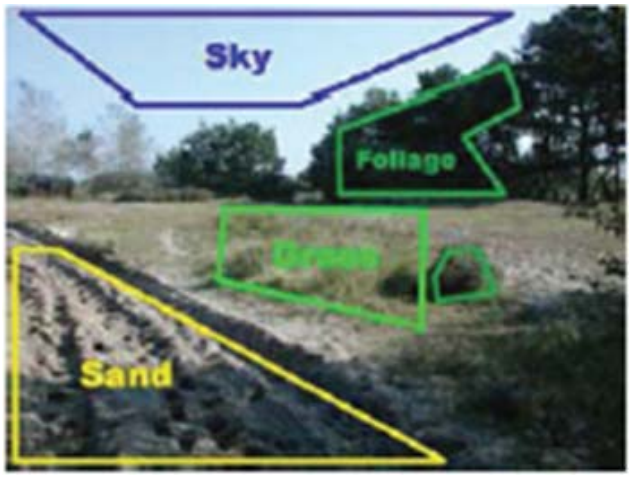

Figure 2-6: Terrain types for each of which Gaussian Mixture Models are constructed [45]

Jansen et al [45], tried to classify road regions by using Gaussian Mixture Models (GMM). This algorithm assumes that images who are taken in similar environments with similar geometry, have similar Gaussian

Mixture distributions in RGB color space. Therefore several GMMs were constructed for terrain types like sand, foliage, sky and grass (Figure 2-6). Then these models were improved by distinguishing the several image sets of the road and using as a training set. Following that, terrain types of a test image, were classified with the GMMs of the environment state which is the most similar to it.

Nefian and Bradski [46] use a Bayesian approach in order to divide unstructured road images into segments and detect the drivable regions within these images. Image models are constructed in order to be utilized while clustering regions like road, road sides, or sky. The clustered areas have several distinctive properties. For example, they all have consistent positions far away from the other regions. However, the sizes and the shapes of these regions display large variations. Embedded Hidden Markov model which is a statistical model builds up models by embedding regions with similar properties is used in this research. By matching the parameters of the model with the upcoming images road regions are tried to be extracted as the one shown in Figure 2-7.

Similarly, Dahlkamp et al [47] utilized GMM in RGB space and tried to adjust this representation to dynamic environments. Moreover, algorithm obtains robust results by combining laser range finder data, color information and pose estimation system. The proposed method is realized by following seven steps. First of all close range road location from sensors, which are varies less with respect to light, are extracted. Then shadow and sky areas are removed from the image. Following that the visual model of near road is learned by the network and then the visual field is voted by the

trained model. After that the identified road patches are selected and finally a drivability map is generated.

Figure 2-7: Extracted road region [46]

Tan et al [48] proposed a system constructing both background and road models utilizing color information and learning. Multiple color models are used for road regions whereas only one model is constructed for background regions. All the models make use of 2D histograms of first two channels from RGB color space representation of the acquired image. Models are updated using the data belongs to the rectangular area just in front of the vehicle. Then, image is classified into road and background regions using these models. Classification is performed by estimating the

probability of each particular model. Finally, temporal fusion, in which road probabilities from several consecutive images are fused with a given methodology, is used in order to alleviate the results.

In a research carried out by Zhang and Kleeman [49], roads are modeled with rapidly adapting 3D color histograms and geometric relations. Then, the road probability image constructed by data obtained via a panoramic camera. After that construction, model was tried to fit to the probability image for the extraction of road regions.

In another study carried out by Zheng et al [50], HSV color space components are used instead of RGB color components. The road region is assumed to have a trapezoidal area and modeled geometrically regarding this property. First of all, a rectangular region, which is just ahead of the vehicle and guaranteed to be road, is selected. HSV color channel values are recorded for a time for this region. Following that, the average color channel values are taken as the road color features and the color image is filtered using these values. The noises in the obtained binary image are eliminated by using basic morphological operations. Finally, the trapezoidal region parameters are obtained using the binary road region. After these parameters are obtained, the road boundaries can be extracted.
To conclude the subsection, it can be said that algorithms belonging to this subgroup have a road model which is tried to converge to the real road regions in the image by using the information obtained via a camera.

2.3.2 Feature Based Methods

In feature based algorithms, by using certain properties of the color road images, distinct features are generated. These properties may be related to color properties, texture properties, frequency properties or other identifier properties. Feature based approach makes use of color and texture differences of the roads to distinguish the road and non-road areas. Even if the performance of this approach is closely related to illumination conditions, the major pro of these algorithms is that, it needs only a little previous knowledge and shapes of the roads do not affect the performance of the algorithm. Hence, it is widely preferred for unstructured road detection.

In a study which is applicable both to structured and unstructured roads, Cheng et al [20] constructed features using the color channels of the road images and then applied mean shift algorithm, which is an unsupervised clustering algorithm, in order to cluster pixels with similar features into road / non-road regions.

In a similar study, Hu et al [51] used color, edge and shape features extracted from road images for classification. Road are gray with a little blue-while regions of non-road are often green and yellow such as plants and dirt, which allows to discriminate roads by color information. Therefore, color features were constructed as the average values and the standard deviation of color channels. Edge features were obtained by using canny edge detector. Finally, road shape feature consisted of the road width. Road width was assumed to have a value greater than a specified

threshold and then the edges inside this width were eliminated. These features were used to cluster pixel into three main regions which are road, non-road and ambiguous regions.

Rasmussen [52-53] used Gabor filters for extracting the texture structure of road images. By using the features containing texture information with a voting procedure, position of the vanishing point, which is the point where the road boundaries are intersected on the horizon, was calculated. By tracking the position of the vanishing point in consecutive frames, the road boundaries were estimated by the algorithm proposed. This algorithm was said to be effective in ill-structured off-road settings.

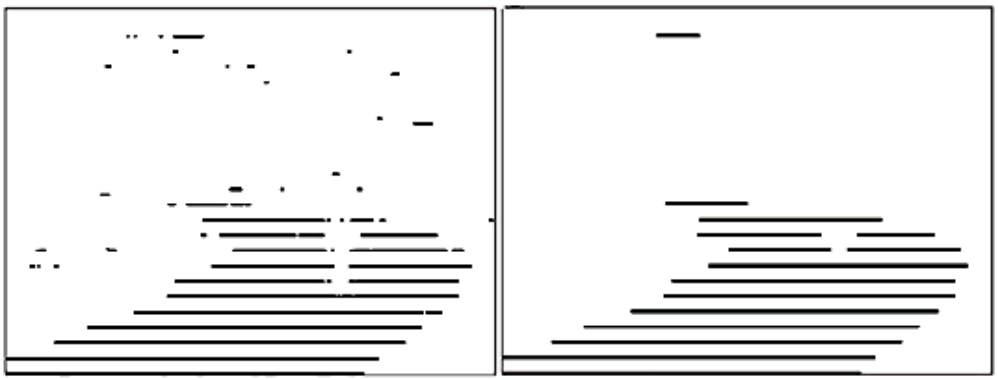

Figure 2-8: Classified and post-processed road regions [54]

Huang et al [54] used HSV color space of the road image and some road properties for generating some features for the classification. Feature related to HSV color space is called color purity, which is a feature

calculated through saturation and value channels of HSV color space. By using this feature, classification is carried out for several region strips of the image in order find road areas. After the classification, small disconnected road regions were merged (Figure 2-8). Following that, by using the parallelism concept of the road boundaries, misclassified regions were eliminated to emerge road regions.

In a study held by Gao et al [55], both RGB and HSV color space was used for the feature extraction. Average values for each channel RGB color space as well as average values for each channel of HSV color space were taken as feature for each image patch. Then these patches were labeled as road (1) / non-road (0) for the learning phase. After the construction of features and labels a rough set technique was used for classification.

In a road detection algorithm governed by Wang et al [43], the road images firstly enhanced by using white balance and gray level stretching techniques. Then, by using an overlapped sliding window, hybrid features were extracted from the image. Hybrid features were composed of three types of features. These are color features, edge features and texture features. Color features are composed of normalized average color channel values. Edge features are composed of the edge image that can be obtained through the procedure given. Texture features are composed of contrast, energy and correlation properties of the image that are defined by the researchers. After the feature variable generation, using support vector machine (SVM) based classification image regions were classified as road or non-road.

Alon et al [56] combined classification via features approach with geometric projection used for recovering Pitch and Yaw in order to identify familiar road regions of the image. The algorithm is composed of region-based and boundary-based components. In the first component, different feature extraction techniques, which are Oriented Gaussian Derivative Filters, Walsh-Hadamard kernels and Moments method namely, are used for feature extraction. After the extraction of these texture features, image was segmented into road and non-road regions by using a classification algorithm named Adaboost. By fusing the results of both components, road regions in the image can be detected.

In a study carried out by Konolige et al [57], a complete autonomous off-road navigation system using stereo vision as the only sensor is designed. System uses a learning algorithm, which is an unsupervised segmentation algorithm. To distinguish different region features, texture and color vectors around small neighborhoods are clustered to constitute up small set of basis vectors which are also called as textons. Color features utilize the RGB values of a 3 × 3 pixels region and for the center pixel 18x1 color feature vector is built. On the other hand, 3 × 3 neighborhood is utilized to represent texture by using pixel intensity gradients with respect to the center pixel. For evaluation of the texton built up descriptors related to the texture, texmos3 texture mosaic, which is included in the Brodatz texture database of University of Southern California, is selected and the descriptors were proved to be valid. Then, these color and texture features, which are extracted from the images, acquired by stereo vision, system clusters similar regions in the image to discover road regions.

To conclude, every road extraction algorithm utilizes features should have features related to the color, texture or illumination properties which are extracted from the color images. After the generation of the feature variables, several classification algorithms are used to classify road and non-road regions in the image acquired.

2.3.3 Machine Learning Based Methods

Machine learning based algorithms utilizes machine learning methods to classify the images into road and non-road (background) pixels. The most common technique is to train a neural network by using features extracted from a color image as input and corresponding image which is labeled as road / non-road as output to this network.

Rasmussen [30] extracted color and texture features from images captured from the on-board camera. All features were extracted for 15x15 regions in the images. Color features are composed of joint histograms of these regions. Texture features are extracted again from these regions using Gabor filters. Other than vision-based features, features were constructed using the laser data. All these features were fed into a two layer neural network for training. After the training phase, when the features of upcoming images are fed as input to the neural network, it classifies the regions as road / non-road.

Similarly, in a study held by Foedisch and Takeuchi [58], a road detection algorithm was proposed, which employed color features derived from color histogram and used them in neural network training. Six possible areas,

three of which are road and three of which are non-road regions, of the images were selected and color features for these windows were generated as can be seen in Figure 2-9. Then, the algorithm was trained using these features with road/non-road labels. Following that, noise appeared in the results was reduced by morphological operations. However, the drawback of the proposed algorithms is that it is applicable only to the roads for which it was trained and cannot be adaptable for other diverse road types.

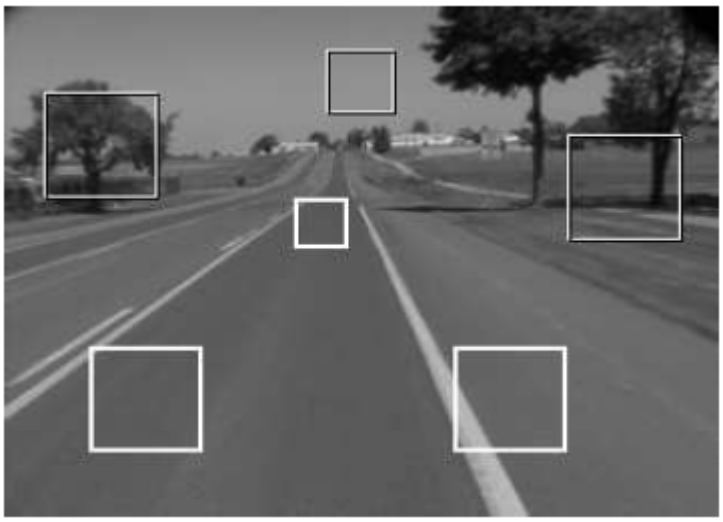

Figure 2-9: Positive (road – white) and negative (non-road – gray) feature windows [58]

Sun et al [59] held a research on unstructured road detection, which utilizes offline training technique, in 2006. In the study for image input, two cameras are used for stereo vision (left and right eye of the vehicle). Then, a

ground projected feature map is constructed for obtaining training data. Feature map is a map in which the features obtained from the images are projected to the real ground terrain. After the feature map is obtained, pixels which are closer than 1.2 meters are labeled as positive (road) and the ones with a distance over 4 meters are labeled as negative (non-road). The remaining portions are labeled as unknown. Road recognition is assumed to be a two-class (road / non-road) classification problem. After the labeling procedure, color features were constructed for positive and negative data sets (image patches). Color feature for each data set is composed of RGB color histogram with a size of 16x16x16 bins. Then, for every bin in each data set probability distribution function is defined and a threshold separating these probability distribution functions is sought for classification through training. The block diagram of learning and classification procedure can be seen in Figure 2-10. After the learning phase, road areas are found through the learned model.

Erkan et al [60] proposed a method utilizing convolutional neural network for road detection. Proposed algorithm depends only on visual data and can adapt environments changing quickly. System is composed of two parts. Firstly a feature extractor is constructed to convert image parts into a lower dimension, which allows algorithm to represent a discriminative problem space, and stereo images are labeled. Then, by using the extracted feature data, a multi-layer ANN is trained. Following that, the features extracted from stereo images are used as inputs to the online ANN that is trained online. While the robot moves on a road, the proposed module outputs the road regions.

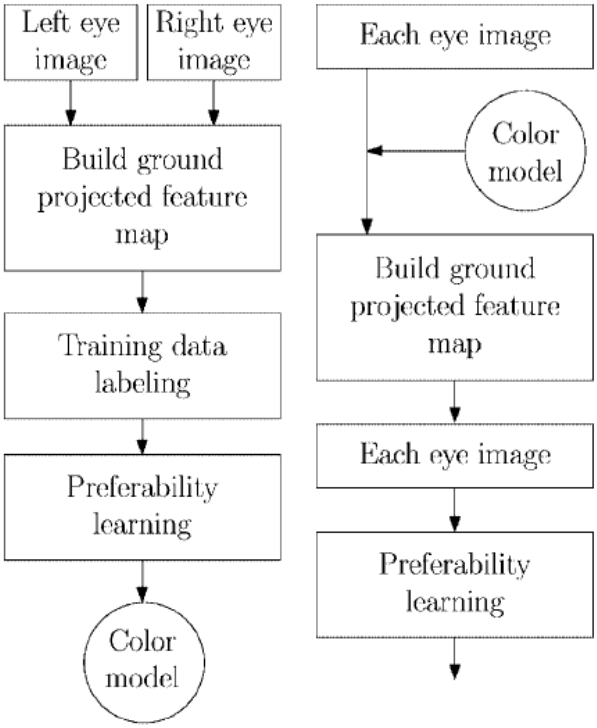

Figure 2-10: Learning block diagram. Learning preferability from example (left diagram). Classifying preferability online (right diagram) [59]

In another study, Hadsel et al [61-62] carried out, stereo vision is used and the features are extracted in a different fashion. Acquired image is split down to patches at several positions. The patches at the upper side which are farthest ones in real world coordinates are shrank down to 12x3 pixel size. By applying this resizing rule, all image patches have the same size. By using this *"distance-normalized image pyramid"* the learning network can be trained most efficiently. First, a road / non-road labels are given to each 12x3 pixel image patch which is extracted from the images acquired by the right camera. Following that, using a set of 120 radial basis functions

(RBFs), feature vectors are computed for each image patch. A feature vector, including features related to color, texture and shape, is constructed with Euclidean distances between a 12x3 RGB window and the 120 fixed rbf centers. Then, a linear learning network is trained using these features as seen in Figure 2-11. Moreover, for every XYZ locations in the world coordinate around the robots, feature information is stored for each patch backward in the timeline using a quad-tree, which is a data structure for spatial information storage. This storage allows the researchers to use the previous data while training the network and is speculated as a short term memory.

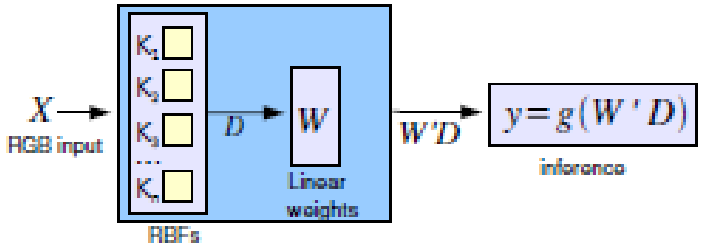

Figure 2-11: Online learning architecture [61]

To conclude the subsection, for algorithms using neural network training, feature vectors are generated using the frames of the video taken off-road where the road/non-road counterparts of the images are marked for learning purposes. Feature vectors are fed as input to the neural network, while the road/non-road labels are fed as output. for neural network training to work well in the detection of unstructured roads, it is evident that videos should be captured under different road and environment

conditions and the training data created from these videos should be fed into the neural network.

2.3.4 Other Methods

In this subsection, algorithms, which do not fall into the previous algorithm groups, are presented. Neither of these algorithms uses features, models or neural network training for extracting structured road regions out of the road images.

Lieb et al [63] proposed an algorithm using reverse optical flow data in order to find road regions. Algorithm works in three steps which are finding road templates via reverse optical flow first, horizontal 1-D template matching and dynamic programming for road location optimization. In the first step, a definition line is selected just in front of the vehicle and by using reverse optical flow data, the location of that definition line in the previous images are found which turns out to be the horizontal templates. After road templates are found, in the second step, these templates are placed over current image and sum of squared differences (SSD) between the image and the templates are calculated. Then, the strongest correlation points are found to describe the center of the road. Due to illumination effects, the strongest SSD correlation is not always the center of the road. In the final step, by minimizing an overall cost function which is the arithmetic inverse of SSD, the center points are updated by dynamic programming and 4th order polynomial is fitted to the points to draw the road center. The widths of the road are found by interpolating the widths of definition line and the template lines.

Li et al [33] proposed a road recognition algorithm, which can be used both structured and unstructured road, utilizes fuzzy reasoning. Road image acquired from onboard camera was decimated down to image with a size of 128 x 128 pixels. After that, the pixels were clustered into either a uniform of a border region to extract edges by using fuzzy logic inference, which were defined by triangular functions such as Min-Max interference and centroid defuzzification. After the edges were extracted, the road region was specified.

Poppinga et al [64] carried out a study about the detection of drivable regions using 3D data obtained via two distinct sensors. These sensors are a infrared camera and a stereo camera. The algorithm constitutes up a parameter space. The parameter space is composed of bins which are specified as drivable and non-drivable. Drivable surfaces are found by examining the number of hits to the corresponding. 3D data is characterized using Hough transform. Several shapes have specified Hough transform outputs. In the parameter space bins regarding to a certain shape are filled by using sample images. After that when a new image comes to the system, it is place to the corresponding bins and the region which does not match with a shape is considered as a drivable region.

Wei and ShuFeng [65] presented a study for detection of unstructured road problem. In this study, first of all, the images are filtered using a median filter in order to reduce the noises in the image. Following that, image is segmented into several using a multi-threshold Otsu algorithm and a two-

peak method. In these algorithms, image has assumed to have L gray levels. Probability of the pixels in i^{th} level (P_i) and the grand average of the image's gray levels (μ_T) are found as designated in equations (2.2) and (2.3) respectively.

Total Number of pixels

$$N = \sum_{i=0}^{L-1} N_i \tag{2.1}$$

Probability of i^{th} level pixel

$$P_i = \frac{N_i}{N} \tag{2.2}$$

Grand average of Image's gray levels

$$\mu_T = \sum_{i=1}^{L} iP_i \tag{2.3}$$

If it is assumed that there are m classes to be separated and m-1 thresholds, namely $k_1, k_2, \ldots, k_{m-1}$, are marked. Then the between-class variance (σ_{BC}) appears in

$$\sigma_{BC} = \omega_0 \left(\mu_0 - \mu_T\right)^2 + .. + \left(\mu_{m-1} - \mu_T\right)^2 \tag{2.4}$$

where

$$\omega_0 = \sum_{i=0}^{k_1} P_i, ..., \omega_n = \sum_{i=k_n+1}^{k_{n+1}} P_i, ..., \omega_{m-1} = \sum_{i=k_{m-1}+1}^{L-1} P_i \tag{2.5}$$

$$\mu_0 = \frac{\sum_{i=0}^{k_1} iP_i}{\omega_0}, ..., \mu_n = \frac{\sum_{i=k_n+1}^{k_{n+1}} iP_i}{\omega_n}, ..., \mu_{m-1} = \frac{\sum_{i=k_{m-1}+1}^{L-1} iP_i}{\omega_{m-1}} \tag{2.6}$$

Then, the separation factor (S) is obtained like in equation (2.7) where the total variance (υ_T) is calculated in equation (2.8).

$$S = \frac{\sigma_{BC}}{\upsilon_T} \tag{2.7}$$

$$\upsilon_T = \sum_{i=0}^{L-1} \left(i - \mu_T\right)^2 P_i \tag{2.8}$$

By using these relations, the image is tried to be clustered. Via an iterative manner until the desired S value is obtained, the calculations are performed and finally a threshold is obtained. After that part using Canny operator the boundaries of the road is extracted.

In conclusion, the algorithms belong to the other methods subsection do not use one of the methods specified in the previous three subsections directly but they utilizes some other uncommon, more authentic methods for road detection.

2.4 Data Generation Methods

In machine learning applications and in other classification methods, generation and preparation of training data is essential. Training data generally is composed of features extracted from an organized data and

positive or negative labels corresponding to that features. Generating and organizing training or classification data is a very-time consuming process when it is performed manually. There is a very limited number of training data generation methods available in the literature, none of which is for the neural network training applications used for road detection.

In another study, Ishida et al [66] proposes a method for training data generation for the recognition of traffic sign symbols. Paper presents that, for accurate detection of traffic sign symbols from images, even from degraded images, the training scheme should be fed by degraded images. In the system using a camera fixed on a car, images containing traffic sign symbols are collected and then method degrades the collected images by using several degradation models and a genetic algorithm. Instead of seeking degraded images exhaustively, method proposes to generate the degraded images automatically.

In one study, great variety of training data is required for sign language recognition system. For that purpose, Jiang et al. propose an automatic data generation scheme for increasing the variety of training data by utilizing the mean shift procedure and normalization [67]. Similarly, in a study examining the hand sign recognition, an automated hand data generator tool is proposed by Yang et al [68], which extracts hand data from video and groups them according to the hand shapes and then allow user to select hand candidates among the data generated. A preview of the graphical user interface of the generator tool is illustrated in Figure 2-12 .

In some other study, a training data generation method is proposed from Magnetic Resonance Imaging (MRI) images for brain tissue classification. The procedure uses feature space proximity measures and does not make any assumptions about the tissue intensity distributions ant it is robust against morphological deviations from the model [69]. In a study having similar interest like the previous study, brain tissue classification is aimed to be accomplished by using a computerized training data generation scheme. According to tissue probability maps, spatial locations of tissue prototypes are selected autonomously, without requiring a human interference [70].

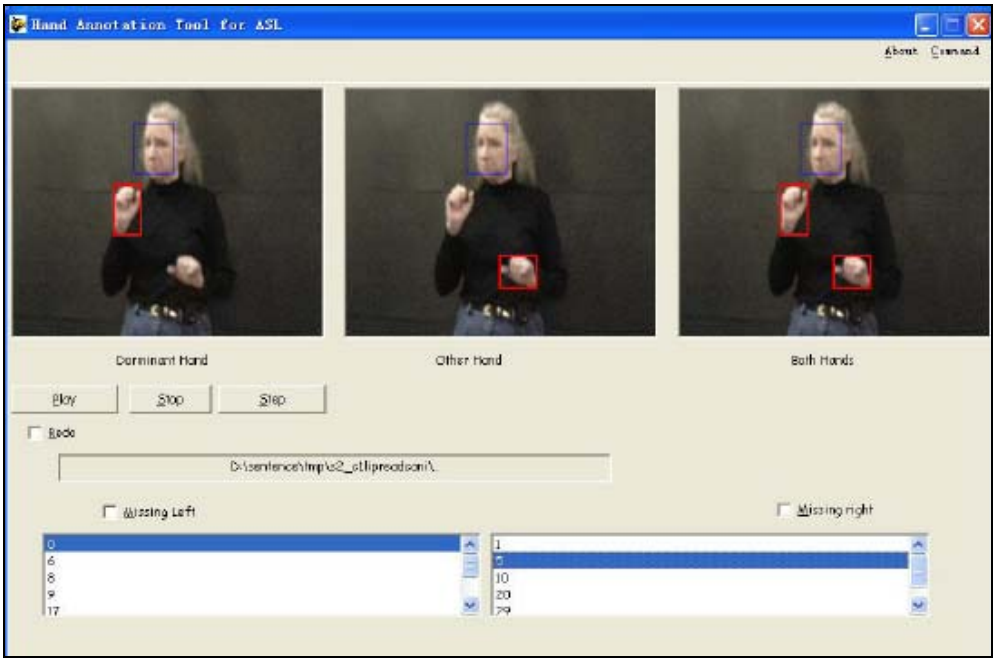

Figure 2-12: Automatic hand data generator - GUI [68]

In a different study, a model is offered for synthetic training data generation for a Hidden Markov Model (HMM) based handwriting recognition system. In this algorithm, several perturbation models are introduced to handwriting data in order to vary it. Perturbation models, which are applied to the data, are thinning/thickening and geometrical transforms. In thinning/thickening operations, thicknesses of handwriting strokes in data are altered using morphological operations. In application of geometrical transformations, the straightness of horizontal and vertical handwriting directions are distorted as well as the connectivity characteristics of letters are altered [71]. Similarly, in another study, a great variety of training data is required for activity monitoring and behavior analysis. In this study, by using statistical techniques, amount of training data is increased while the newly generated data is given a variation using several perturbation models [72].

To summarize the subchapter, even though task-specific data clustering methods exist in the literature, a more generic clustering method to generate neural network training data is not available in the literature.

CHAPTER 3

DATA GENERATION FOR ARTIFICIAL NEURAL NETWORK TRAINING

3.1 Introduction

Dealing with mobile robots inevitably forces one to process images so that the robot can take proper action within its surrounding. Yet, the rich content of imagery present around the robot, especially outdoors, enforces one to process image sequences from a video stream rather than a single snapshot. This inevitability is the driving force of a data generation method for ANN (ANN) training. Processing video streams to segment images for training purposes is a thorough process; hence, in this chapter, a simpler method developed for generating such training data is presented.

Traditional data selection out of a road image depends on the manual segmentation of road areas by selecting road boundary (Figure 3-1). The proposed method lets the user mark quadrilateral areas of interest in a video stream in an online fashion. These segmented image sequences serve as ANN training data and in comparison to manual segmentation, the proposed method offers an outstandingly shorter time. The proposed method is referred to as "*Quadrilateral On Screen Segmentation (QOSS)"* and as proposed, it nicely fits the problem of extracting road boundaries and areas from a video stream. However, QOSS does not only provide a specific

solution to road detection, but it can also be used in numerous different applications as will be speculated at the end of this chapter.

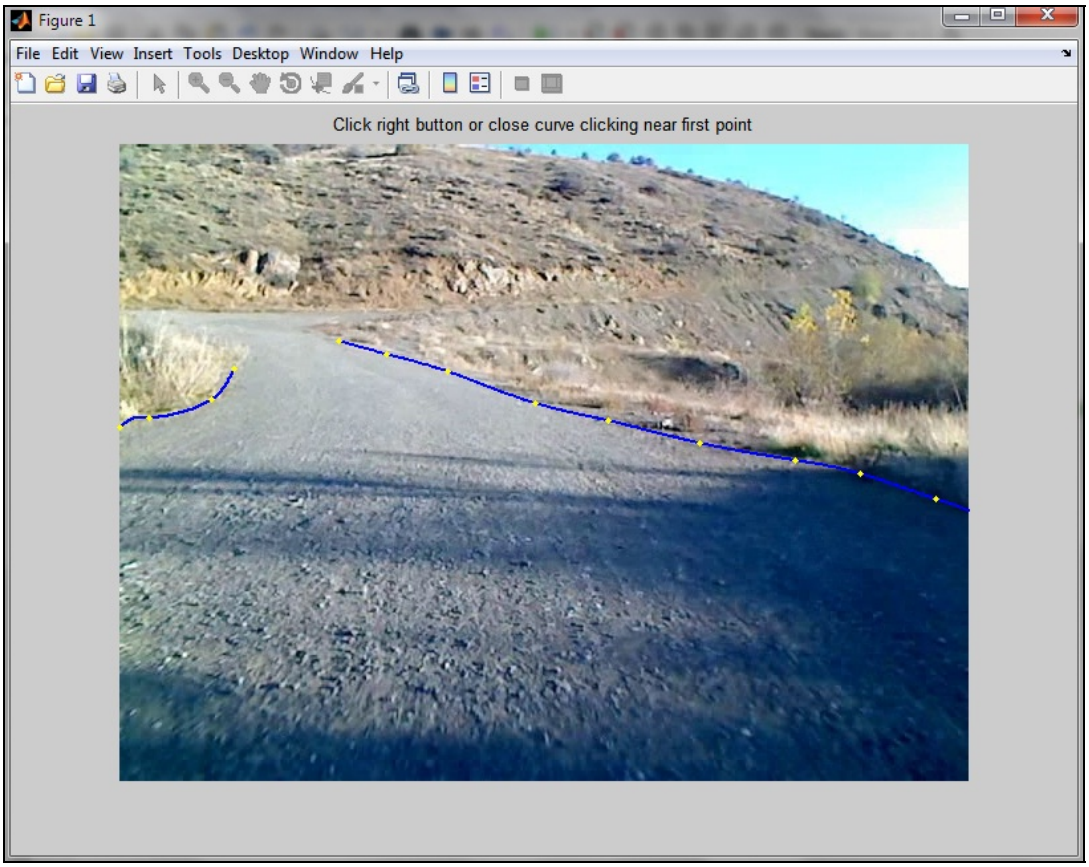

Figure 3-1: Traditional data generation

The QOSS process implies processing the video of another video. The video rehabilitate file to be processed (in our case, *road video*) plays on a PC monitor where a camera facing the monitor records a second video, which will be referred to as *fingertip video* in the rest of the chapter. While the

original video plays on the monitor, a user wearing a black glove with marked index fingers and thumbs moves his/her fingers so that area of interest is contained within the quadrilateral area formed by his/her fingers. Once the video recording is completed, the road regions are segmented in the rest of the QOSS process by finding the locations of the finger markers on the fingertip video and transforming these coordinates into the original video coordinate system.

In the remaining section, the proposed method will be presented in four parts; the setup of QOSS, the detection of selected road points on screen, the transformation of these points into road video coordinates and finally the designation of the road/non-road areas on this video.

3.2 The QOSS Setup

The QOSS setup consists of three main parts (Figure 3-2). These are a simple camera, a PC monitor on which the road video is played and a pair of marked gloves worn by the user. In our application of the QOSS, a Logitech Quickcam Pro9000 camera is used. The camera mounted on a tripod is placed in front of the monitor. The world line of the camera should be nearly perpendicular to the screen to eliminate the need of 3D transformation of the coordinates, yet no extreme effort is needed to guarantee this orthogonality. The vertical orientation of the camera is not vital since by using a 2D transformation, the selected road coordinates on the fingertip video will be transformed to the coordinates of the road video. Black gloves with red markers on the fingertips are used to select the four corners of an area of interest within the road region.

3.3 The Detection of Selected Road Points (SRP) in QOSS

In the QOSS process, the locations of fingertip markers are found and using the location of the finger tips, areas of interest in the road region are extracted. Considering the fact that the QOSS process takes place in a lab environment, where the lighting conditions can be easily controlled, white balance and contrast values of the camera are set to fixed values so that the fingertip markers become more apparent, and hence their detection becomes easier and faster.

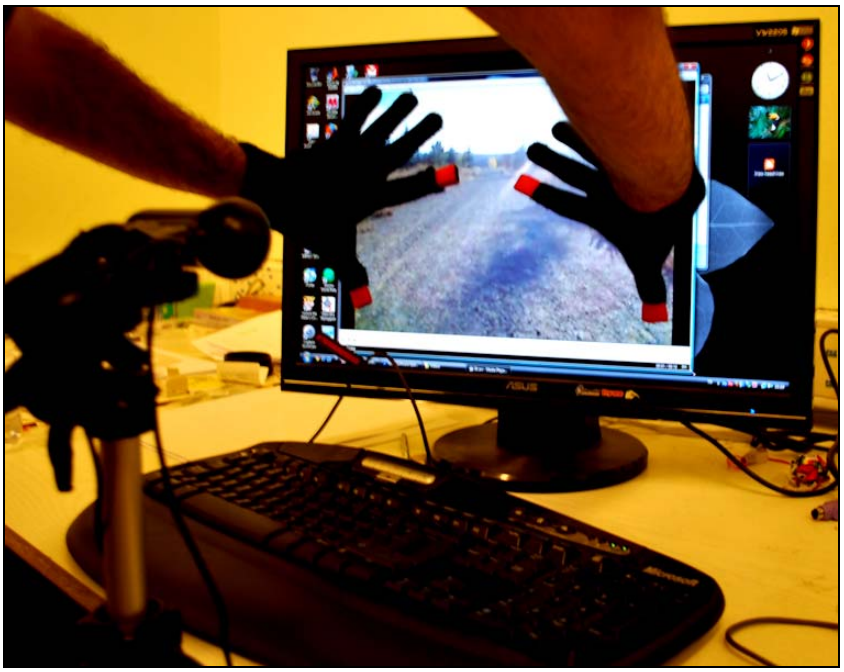

Figure 3-2: QOSS setup

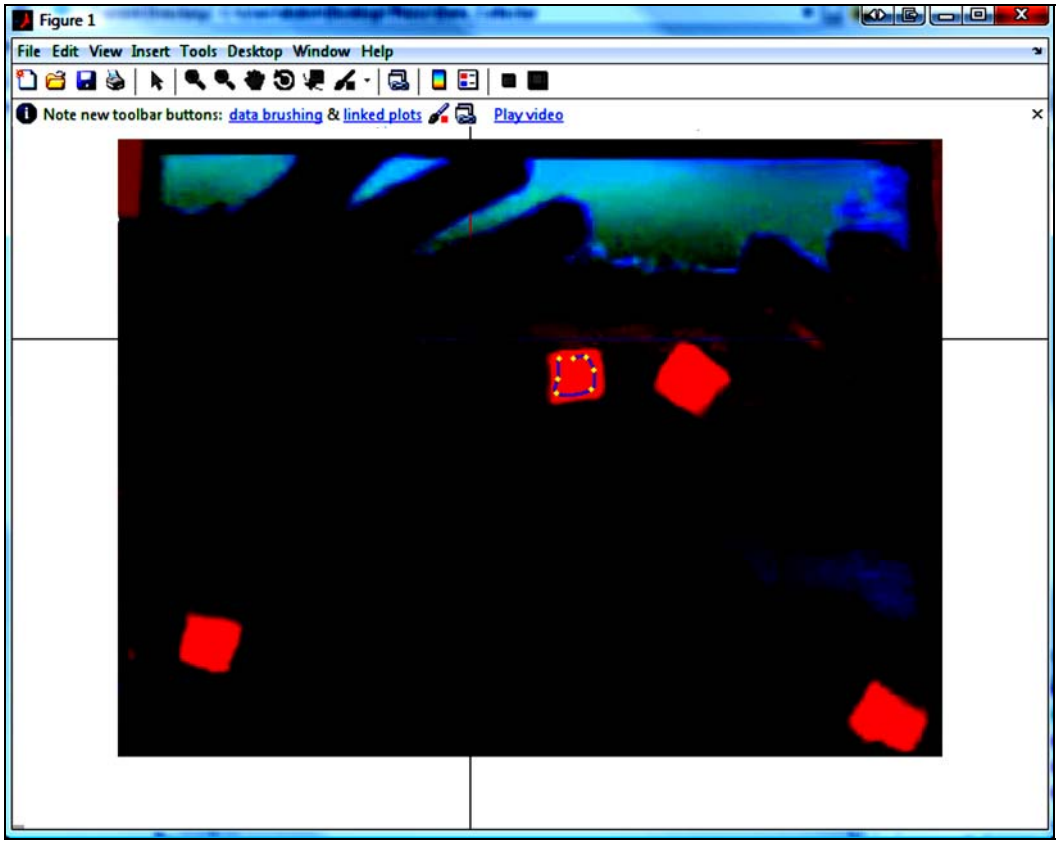

Figure 3-3: Determination of RGB color bands for detection

Once the white balance and the contrast values are set, the next step is the consistent detection of the fingertips using the Red-Green-Blue (RGB) color range of the video stream images. At this point, the user is asked to select the fingertip markers on a sample frame similar to the one shown in Figure 3-3. After selecting the four markers, a RGB value interval is determined automatically. The center of regions found in each image frame based on this RGB interval are calculated and the locations of the markers are recorded along with the processed image frame for the upcoming steps.

Due to the illumination effects mainly caused by the user (i.e. shadow of the arm that falls on the markers at the bottom), discontinuities might be introduced, and hence the number of regions found in an image may be more than four. In this situation, the centers of the regions are calculated once more and then these centers are clustered into four positions by k-means clustering technique since it is known that in the end we should have four marker positions.

3.4 The Transformation of SRP on Real Road Video

While recording the fingertip video, the camera may be tilted; hence, the camera image plane and the monitor surface might not be parallel to each other. Moreover, the boundaries of the playing road video may not exactly fit the viewfinder of the recording camera. Therefore, after the extraction of the positions of markers on the fingertip video, these positions should be transformed into the coordinate frame of the road video. For achieving this coordinate transformation, Homography transformation is used. Under homography, in the image planes, using the 2D homogeneous coordinate the transformation is depicted as in equation (3.1) [73].

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \underline{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.1}$$

where $\underline{H}$ is the homography matrix and λ is a scalar.

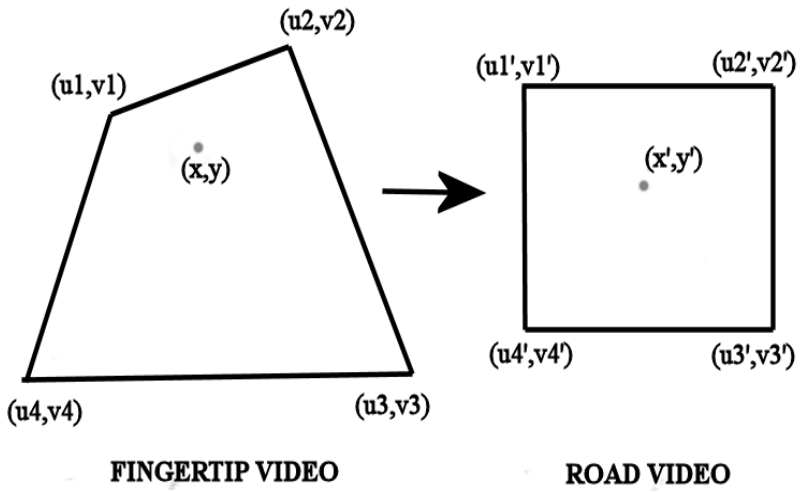

Figure 3-4: Homography transformation

For transformation, homography matrix should be extracted by using the data on the recorded video. In the application, the user is asked to select the four corners of the playing video. Therefore, four points from the first video and its counter parts in the second video are known (Figure 3-4). By using these coordinates in equation (3.1) and accepting H_{33} element of homography matrix as 1, twelve equations are obtained as in equations (3.2) to (3.13).

$$u_1 H_{11} + v_1 H_{12} + H_{13} - u_1' \lambda_1 = 0 \tag{3.2}$$

$$u_1 H_{21} + v_1 H_{22} + H_{23} - v_1' \lambda_1 = 0 \tag{3.3}$$

$$u_1 H_{31} + v_1 H_{32} - \lambda_1 = -1 \tag{3.4}$$

$$u_2 H_{11} + v_2 H_{12} + H_{13} - u_2' \lambda_2 = 0 \tag{3.5}$$

$$u_2 H_{21} + v_2 H_{22} + H_{23} - v_2' \lambda_2 = 0 \tag{3.6}$$

$$u_2 H_{31} + v_2 H_{32} - \lambda_2 = -1 \tag{3.7}$$

$$u_3 H_{11} + v_3 H_{12} + H_{13} - u_3' \lambda_3 = 0 \tag{3.8}$$

$$u_3 H_{21} + v_3 H_{22} + H_{23} - v_3' \lambda_3 = 0 \tag{3.9}$$

$$u_3 H_{31} + v_3 H_{32} - \lambda_3 = -1 \tag{3.10}$$

$$u_4 H_{11} + v_4 H_{12} + H_{13} - u_4' \lambda_4 = 0 \tag{3.11}$$

$$u_4 H_{21} + v_4 H_{22} + H_{23} - v_4' \lambda_4 = 0 \tag{3.12}$$

$$u_4 H_{31} + v_4 H_{32} - \lambda_4 = -1 \tag{3.13}$$

These 12 equations are enough to solve 12 unknowns that consist of eight unknowns from homography matrix and 4 unknowns from scalar values (λs). These equations can be represented in a matrix form as in equation (3.14).

$$\bar{Q}\bar{x} = \bar{P} \tag{3.14}$$

where

$$\bar{x} = \begin{bmatrix} H_{11} & H_{12} & H_{13} & H_{21} & H_{22} & H_{23} & H_{31} & H_{32} & \lambda_1 & \lambda_2 & \lambda_3 & \lambda_4 \end{bmatrix}^T \tag{3.15}$$

$$\overline{Q} = \begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & 0 & 0 & -u_1' & 0 & 0 & 0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & 0 & 0 & -v_1' & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & u_1 & v_1 & -1 & 0 & 0 & 0 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -u_2' & 0 & 0 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & 0 & 0 & 0 & -v_2' & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & u_2 & v_2 & 0 & -1 & 0 & 0 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -u_3' & 0 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & 0 & 0 & 0 & 0 & -v_3' & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & u_3 & v_3 & 0 & 0 & -1 & 0 \\ u_4 & v_4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -u_4' \\ 0 & 0 & 0 & u_4 & v_4 & 1 & 0 & 0 & 0 & 0 & 0 & -v_4' \\ 0 & 0 & 0 & 0 & 0 & 0 & u_4 & v_4 & 0 & 0 & 0 & -1 \end{bmatrix} \tag{3.16}$$

$$\overline{P} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & -1 \end{bmatrix}^T \tag{3.17}$$

Therefore, homography matrix can be extracted by using equation (3.18).

$$\overline{x} = \overline{Q}^{-1}\overline{P} \tag{3.18}$$

After the homography matrix is extracted, the location of the road corners in road video can be obtained by multiplying the homography matrix with the marker coordinates. The result is a 3x1 vector. To obtain the correct (x′,y′) coordinates, the resulting vector is divided by its third component like in the example in equation (3.19).

$$\begin{bmatrix} x_{result} \\ y_{result} \\ c \end{bmatrix} \Rightarrow \begin{bmatrix} x_{real\_result} \\ y_{real\_result} \\ 1 \end{bmatrix} = \begin{bmatrix} x_{result} / c \\ y_{result} / c \\ 1 \end{bmatrix} \tag{3.19}$$

A sample projection results obtained through the application can be seen in Figure 3-5.

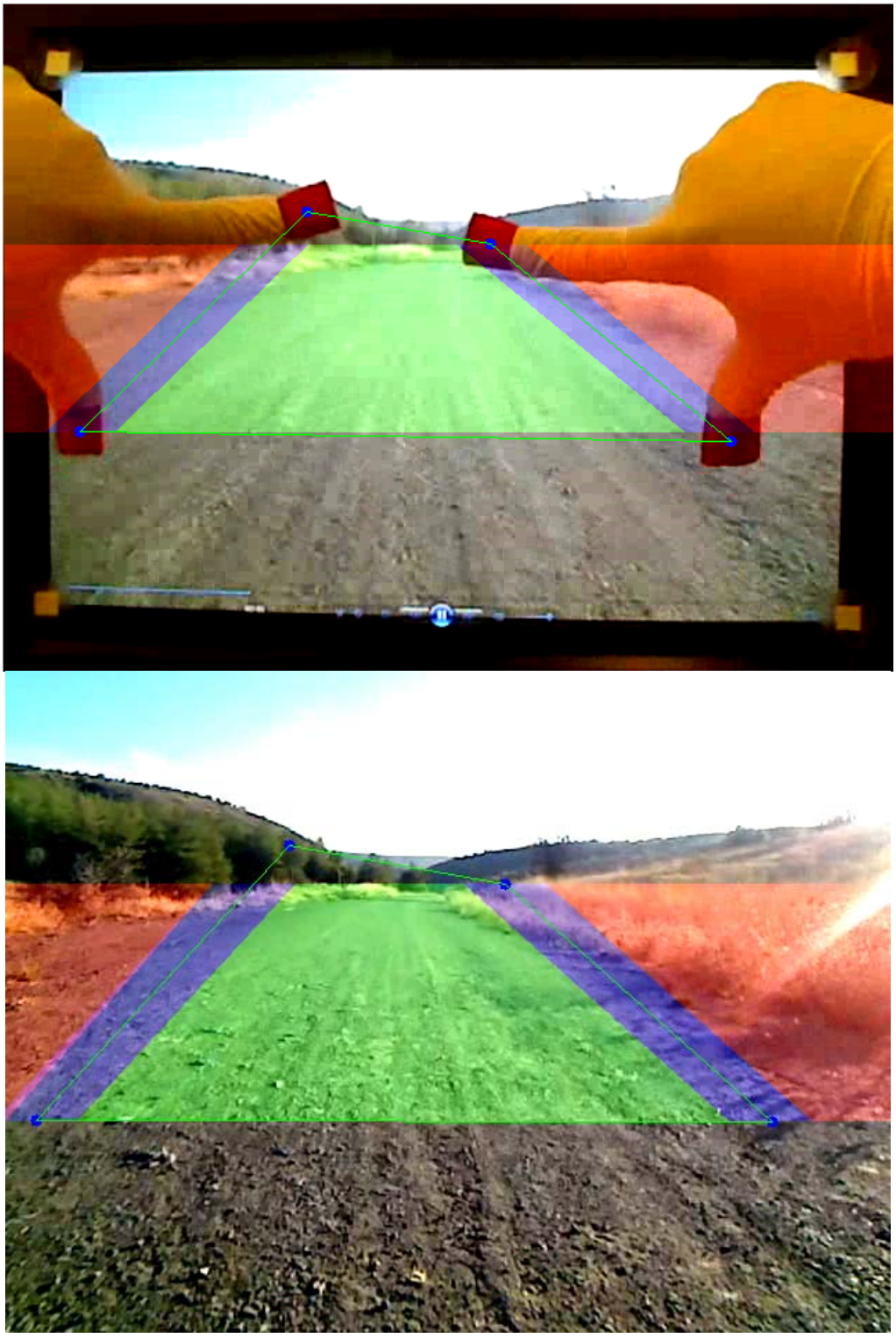

Figure 3-5: (a) Real world coordinates (b) Projected coordinates

3.5 The Designation of Areas to be Used in Training

Once the marker coordinates are projected from the coordinate frame of the fingertip video to that of the road video, the road and non-road parts of the video are to be labeled accordingly.

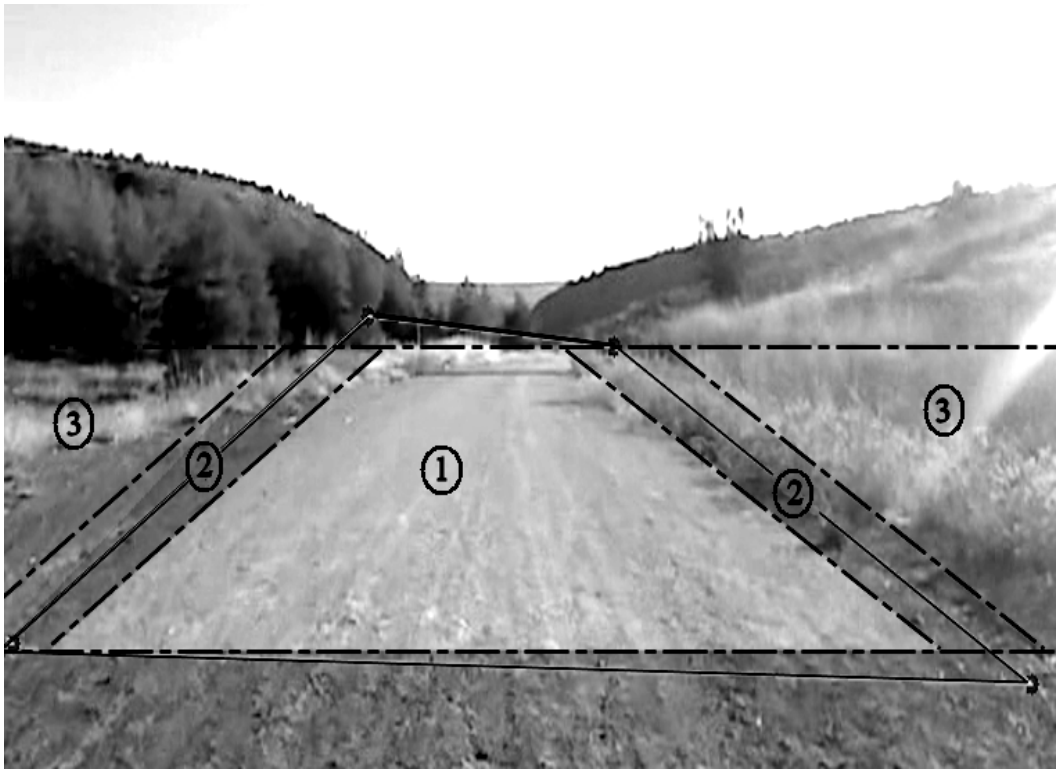

Figure 3-6: The projection of the coordinates in real road video frames. (1)Road region - (2) Ambiguous region - (3) Non-road region.

As illustrated in Figure 3-6, by the use of thin solid lines, the transformed marker positions define a quadrilateral from which a parallel strip is selected as the area of interest. The selected area is then labeled as road,

ambiguous, and non-road as illustrated in Figure 3-6. Recalling that this work depends on a trade-off between speed and precision, the sides of the quadrilateral area do not necessarily separate the road and non-road regions with a clear cut. Hence, an ambiguous region is defined around the sides of the quadrilateral area as a safety zone, the width of which can be defined by the user.

In the QOSS method, during the training data generation, the whole image extracted from the road video is not processed; instead, only the portion of the image in the safe band is processed. Therefore, the time spent in training data generation using the QOSS method is much less than the normal processing time.

3.6 Experiments for the Evaluation of the QOSS Method

QOSS, as explained above, offers a faster method of clustering video streams in comparison to manual segmentation methods. Yet, this method only lets the user select a subset of pixels that can be selected by using manual methods. Therefore, before the effectiveness of QOSS can be claimed, the usability of data created via QOSS should be justified. For this purpose, a set of experiments are conducted, where a set of videos are segmented both manually and via QOSS. A subset of the segmented images is then used to train ANNs, where the remaining of these images are used to assess the learning performance. After the experiments are completed, the performance of ANNs trained via QOSS and data that is manually trained are compared.

To be used in these experiments, videos were recorded on unstructured roads around the lake in Middle East Technical University Campus by using an All-Terrain Vehicle (ATV) in April of 2008. 3 videos with VGA resolution (640x480) were captured via a webcam attached to the front of the ATV while driving the vehicle at speeds in the range of 8-25 km/h. The shooting session took approximately 37 minutes in late morning and a total of 49,723 video frames were recorded in the end.

For training, 24 video sections containing 300 frames are cut with different illumination and road texture characteristics from each of the 3 videos resulting in 24 videos. Out of each 24 video, 30 frames were randomly selected, producing 720 frames for training purposes.

During manual labeling, road and non-road regions of these 720 images were labeled by selecting the road area by defining the polygon containing the road area. Road areas were marked with 1's whereas non-road areas are marked with 0's. The labeling of each frame took approximately 1 minute. With the best estimate, the generation of ANN output data took approximately 12 hours.

In the QOSS labeling process, road, ambiguous and non-road areas are marked with 1's, 0.5's and 0's respectively. As in the case of manual labeling, videos are not down samples and all 7200 frames are labeled since the labeling operation is performed online as the road video plays. The time spent during the labeling equals to the total duration of 24 videos, which is around 4 minutes. After labeling, the same 720 road images are selected automatically for comparison.

The labeling process creates the reference output values for ANN to be used during training. Inputs to these networks are given in the form of RGB feature vectors. These vectors are generated from each of 15 x 15 pixel-sized image patches. On the images that are manually segmented, a safety strip of 20 pixels is left, and feature vectors are computed at 10-pixel intervals vertically and horizontally within the remaining region (Figure 3-7). Feature extraction from images labeled via the QOSS method is done over the parallel strip as seen Figure 3-6 that is defined by the fingertips.

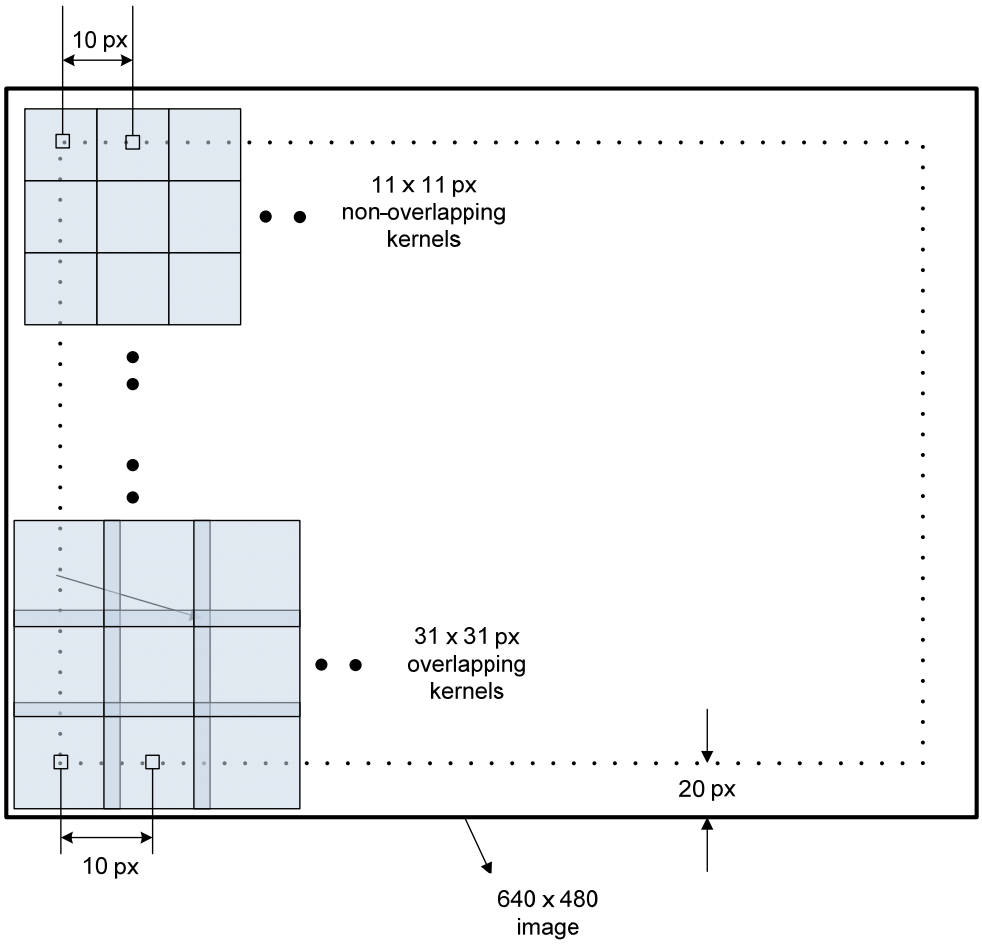

Figure 3-7: Different sized kernels located on a 640 x 480 image

RGB feature vectors used in this experiment consist of three parts: an 8-bins-per-channel independent RGB color histogram, a 4-bins-joint RGB color histogram part and pixel locations of kernel centers, for which the feature vector is created.

The first part of the feature vector is an 8-bins-per-channel independent color histogram [74]. For each color channel, the intensity value is separated into 8 equal parts and the number of pixels falling in each part is computed, resulting in a 24x1 vector. The second color feature is a 4-bins-per- each RGB channel joint color histogram [75]. This can be depicted as a cube whose x-y-z axes are RGB channels and each channel is divided into 4 equal portions on each axes. Hence, the RGB color space is divided into a total of 4^3 bins producing a 64x1 vector to form the second part of the feature vector. At the end, pixel location of the kernel centre (2x1) is combined with the previous vectors to form the 90x1 feature vector.

This feature vector is calculated both for the conventional method and the QOSS method. For the conventional method, the whole image, which is 640x480 pixels in resolution, is used resulting in 2,745 feature vectors per image and 1,976,400 feature vectors for 720 training images. On the other hand, for the QOSS method, feature vectors are extracted only for the selected region. Approximately the half of each image is used for feature extraction resulting in 1,300 to 1,600 feature vectors per image, and a total of 1,001,234 feature vectors are formed for 720 training images.

For the output part of the training data, labeled images are divided into 11x11 pixel sized portions at the same centre pixel locations as the input

feature vectors are extracted. Although the feature vectors in the input side uses 15x15 pixel kernels, here 11x11 pixel kernels are used to avoid the overlapping of the same road/non-road labels. Each portion constitutes up a 121x1 vector which will be fed into the output part of the ANN.

The Matlab Neural Network Toolbox is used to train the ANNs in this work. Each ANN has one hidden layer. The number of neurons in the hidden layer is generally suggested to be between the numbers of input and output neurons. Having input layer of size 90 (same as the size of the input feature vector) and output layer of size 121 (same as the size of road/non-road labels for each kernel); therefore, the number of neurons in the hidden layer is selected as 110. Neural ANN weights are updated using conjugate-gradient back-propagation with the “unipolar sigmoid" activation function. 80% of any given data set was used for training, 10% for testing and 10% for validation during the training phase.

3.7 Results

After training separate ANNs with data generated by manual segmentation and QOSS, each ANN is tested using 50 images which are extracted from the road video. These test images are manually labeled. Feature vectors are extracted from these images and are fed as input into each trained ANN. An example result for both ANNs using the same test image can be seen in Figure 3-8. It can be seen that the results are very similar.

From a mobile robotics point of view, we are only interested in the correct and incorrect detection of road pixels. To serve this purpose, the

performance of each ANN is evaluated based on two measures. **Measure-A** is the percentage of the correctly detected road points and **Measure-B** is the percentage of the incorrectly detected road points. Measure-A and measure-B are computed for all test images.

The ANN trained with the data set, generated via manual segmentation, found 78.6% of the roads correctly with a standard deviation of 30.8%. The best performance of this method found all road points correctly in 3 test images, whereas the worst performance appeared in one image with 6.8% correct detection. ANN trained with data generated via QOSS found 76.6% of the roads correctly with a standard deviation of 26.6%. At best, this method found all road points correctly in 4 test images, whereas at worst, 5.2% correct detection rate was recorded. Hence, in terms of measure-A, no significant change in performance was detected (Figure 3-8).

The results for measure-B are very similar to that of method-A. ANN trained using manually segmented data set found 23.6% of the roads incorrectly with a standard deviation of 31.8%. ANN trained with data generated via QOSS found 27.7% of the roads correctly with a standard deviation of 28.7%. All results are tabulated in Table 3-1.

Figure 3-8: Road detection results of ANNs trained by data generated by (1) classical method (2) QOSS method.

Table 3-1: The comparison of QOSS with the conventional method

		Conventional method	QOSS
% of correctly found road points	**Mean**	78,6	76,6
	Standard Deviation	30,8	26,6
% of incorrectly found road points	**Mean**	23,6	27,7
	Standard Deviation	31,8	28,7
Time spent	**During data generation**	720 mins	4 mins
	During training	32 hours	19 hours

As can be seen from Table 3-1, both ANNs, one which is trained with manually segmented images, the other trained with QOSS, exhibit similar performance. Another important point is that nearly 12 hours were spent during the data generation phase for manual segmentation, while by using the QOSS method, only 4 minutes are needed for the same operation. The QOSS method takes 180 times less time for data generation with a minor difference in road detection performance in comparison to the manual segmentation method. Moreover, an average person cannot label road data with the same concentration for 12 hours straight. Therefore, time spent during manual data labeling would take much more time than the simply computed time. On the other hand, data selection using the QOSS method would take as long as the duration of the segmented video.

Furthermore, since the number of feature vectors extracted from manually segmented images is approximately two times the number of feature

vectors extracted from images segmented via QOSS, the ANN training durations are also different. By using the QOSS method, similar results are obtained by reducing the time spent for ANN training nearly 40% when compared to the conventional method (Table 3-1).

As a result, the QOSS method presents similar road detection results to those of the conventional method, while reducing the time spent during training data generation at least 180 times and the time spent for ANN training by nearly 2 times.

3.8 Conclusion

In this chapter, a practical method, referred to as QOSS, which is used to label video streams, is presented. This method lets the user segment a video stream concurrently as the video plays; hence, this way of segmentation turns out to be at least a couple of hundred times faster compared to manual segmentation. The usability of QOSS is investigated in road detection. ANNs are trained with images that are segmented manually and via QOSS. After completing the training, very comparable detection results are obtained for both methods. The slight under performance of the QOSS method is acceptable and QOSS proves to be a very promising alternative to manual segmentation of videos for various applications.

Even though video streams are segmented for training ANNs to detect road sections, QOSS has a much richer potential use. In this chapter, the proposed QOSS approach is used to dynamically select quadrilateral areas in a streaming video. In a similar fashion, lines (skyline, border, etc) and points (multiple objects, faces etc.) can also be selected following this

approach. In other words, the QOSS approach can be used to select any type of “region of interest” from video data.

CHAPTER 4

PROPOSED TECHNIQUES FOR ROAD RECOGNITION

4.1 Introduction

In this chapter, two approaches are proposed for road recognition. First one is about computing a vanishing point location using the texture properties of the images. Second one is about training different ANNs utilizing different feature vectors extracted from images again. Using the results of these two approaches a reference robot heading, which is expected to lead the robot to follow the road, will be computed. Reference heading computation will be explained in Chapter 5.

First of all, a brief introduction to ANN training is presented at the beginning of the chapter. Following that, vanishing point computation approach is explained in detail and then the best configuration for this approach is presented. In the final subchapter, a thorough explanation for the approach using ANN training is given. In this subchapter, a detailed walkthrough for feature vector generation is presented and at the end of subchapter, the most successful configuration for the training approach is selected. Best configurations for both vanishing point and ANN training approaches will be utilized for reference robot heading computation.

4.2 Introduction to Artificial Neural Network Training

An artificial neural network (ANN) is an organization which is inspired by working principles of biological neurons. That is to say, it is biological neural network emulation. Artificial neural networks appeared after McCulloch and Pitts [76] established simplified biological neuron models, which were used as theoretical parts of circuits performing specified missions, in their paper "Logical Calculus of the Ideas Immanent in Nervous Activity" in 1943.

Biological neuron is modeled functionally with three basic components. Synapses of the biological neuron are represented as weights in functional model. If the signal is passing through specified synapses much more compared to the others the connection in that synapse should be stronger. This property is modeled with the weights whose values represent the strength of a connection between an input and a neuron. A negative value of weight represents repressive connections and a positive weight value shows excitant connections. There are two other parts which simulated the inner neuron cell activity. These are an adder and an activity function. An adder sums all of the inputs to the neural after multiplying them with their corresponding weights. An activation function is in charge of the neuron output state and keep the output within an acceptable range, which is generally between 0 and 1 [77]. A simple mathematical neuron model scheme can be seen in Figure 4-1.

A neural network should be configured in order to obtain desired outputs with feeding the corresponding inputs. For the desired results, weight

should be specified correctly. Setting weights may be done using a *priori* knowledge or simply training the ANN by simultaneously feeding matching input and output patters to the network until the weights converge to some value according to some learning schemes.

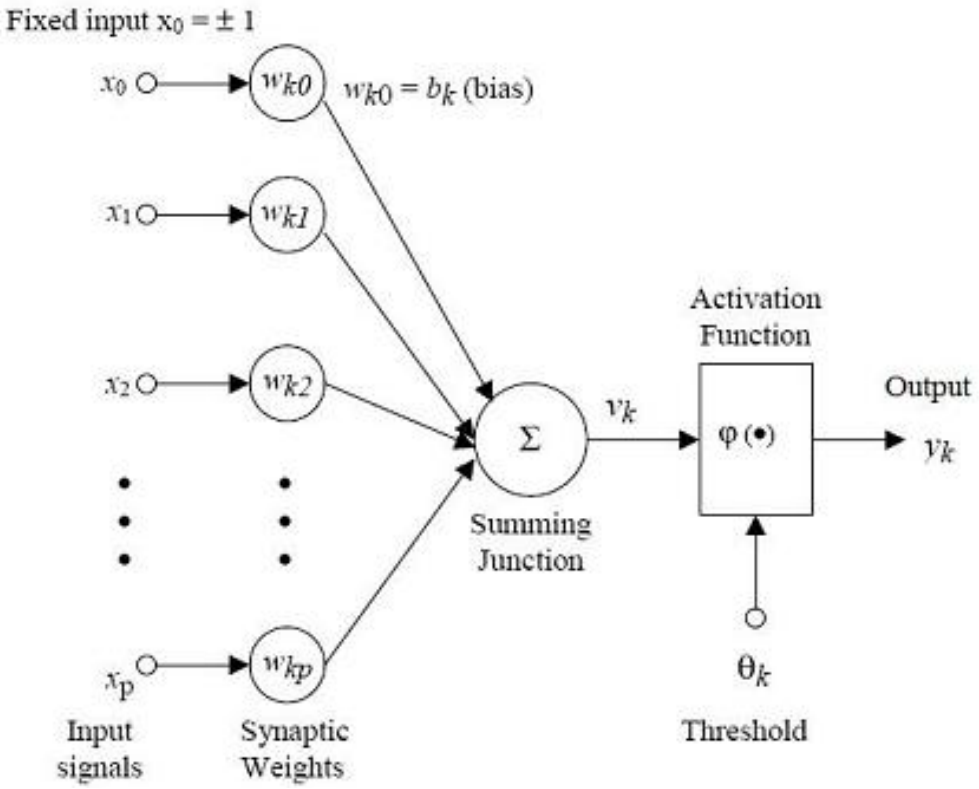

Figure 4-1: A mathematical model of an artificial neuron

Learning schemes can be collected under two main categories. These are supervised learning and unsupervised learning. Supervised learning is a learning scheme in which the ANN is trained by being fed of the matching input and output sets. Unsupervised learning scheme is a scheme in which ANN is trained to respond clusters of patterns within the input set. The

corresponding output set cannot be fed to the ANN in this scheme since there is not an output classification. ANN classifies the separate clusters of input and responds according to these clusters.

Moreover, there are two connection patterns for ANNs. These are feed-forward ANNs and recurrent ANNs. In feed-forward ANNs, information flow from input layer to output layer is straightforward. There is no backward feeding within the network. However, in recurrent ANNs, there are feedback connections.

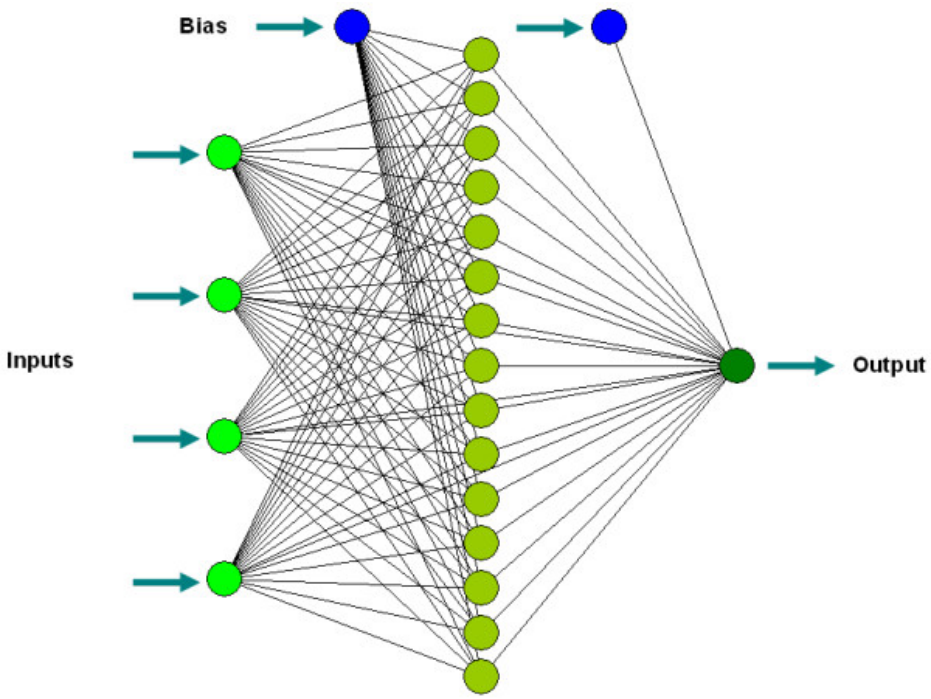

Figure 4-2: Feed-forward ANN[1]

[1] http://www.biomedcentral.com/content/figures/1472-6750-7-53-2-l.jpg

A feed-forward ANN is an ANN where connections between the units do not form a directed cycle. Using feed-forward ANNs, any non-linear relationship can be simulated and applied universally. There is an input layer, hidden neuron layer and output layer in feed-forward ANNs as seen in Figure 4-2. They are used in several Pattern and Object Recognition applications

In this study, the most common ANN configuration used in the literature is utilized. A feed-forward ANN is used with a supervised learning scheme. Back propagation algorithm is used for the selected feed-forward ANN scheme during the training of network. The activation function is selected as 'unipolar sigmoid'.

4.3 Vanishing Point Detection

4.3.1 Introduction

Vanishing point is an attention-grabbing phenomenon and is an motivating research area for vision-related areas since it is practical for several computer vision applications such as obstacle recognition [78], camera calibration [79], road detection and road boundary, curvature estimation [52, 80-82]. When lines parallel in real world are projected into a 2D image plane, they do not seem parallel but seems to converge to each other. Due to perspective, all the parallel lines projected in the image plane met at a point. The meeting point of these lines in the image plane is called Vanishing Point (Figure 4-3).

Figure 4-3: Vanishing point[2]

Especially, in several scenes like roads, pavements, trails etc. there are several parallel lines in image plane converging to one point. Therefore, in road recognition or in road detection algorithms, especially in the ones designed for unstructured road settings having no particular characteristic, vanishing points are used to obtain variety of information about the road and for that reason have a great deal of assistance. In the simplest manner, a vanishing point supplies information about towards where the vehicle on the road should be headed since it represents the end of the road in that scene.

[2]*http://www.pentaxforums.com/forums/attachments/mini-challenges-games-photo-stories/17150d1218807360-project-52-week-14-vanishing-point-vp3-jpg*,Last accessed 22.05.2010

Vanishing point for unstructured road scenes can be detected following the texture orientations in the road image. In unstructured roads, several texture features may lead us to the vanishing point. These can be tire trails lying in the direction of the road and boundary lines of the road even if they may be ambiguous in unstructured roads due to bushes near the road. Using these textures, first of all dominant texture direction will be estimated and then using a voting procedure.

In this chapter, an algorithm utilizing Gabor filters for vanishing point detection will be presented. First, using Gabor filters, dominant texture orientation of the every pixel is obtained in the road image. Then in the next subsection, a voting procedure is described which uses the dominant texture information to find the major candidates of vanishing point. In the following subsection, the experiments and results of this procedure is presented. Finally, a conclusion will be drawn upon the given technique.

4.3.2 Determining the Dominant Texture Orientation

Dominant texture orientation at a pixel position of an image presents the direction in which the strongest texture flow is. Estimating the dominant texture orientation in a road image is crucial for vanishing point detection. Texture orientation can be extracted using Gabor filters.

A Gabor filter is a linear filter used for edge detection in image processing applications. In spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. Gabor filters are self – similar filters since all derivations of these filters are created from one mother wavelet by rotation and dilation. In this study, Gabor wavelet filters

proposed by Sing and Lee [83], since these wavelet filters are proved to be effective practically in several studies [17, 52-53, 78-80, 84-85].

The adopted Gabor wavelet filters are used to extract texture information from image by performing a Gaussian-windowed Fourier analysis on the image. Gabor filters have a specified k x k kernel size and are formulated by filter orientation (θ) and wavelength (λ). There are both odd and even phases of the Gabor filters. Both of the Gabor functions are given in equations (4.1) and (4.2).

$$\hat{g}_{odd}(x,y,\theta,\lambda) = \exp\left[-\frac{1}{8\sigma^2}\left(4a^2+b^2\right)\right]\sin\left(\frac{2\pi a}{\lambda}\right) \tag{4.1}$$

$$\hat{g}_{even}(x,y,\theta,\lambda) = \exp\left[-\frac{1}{8\sigma^2}\left(4a^2+b^2\right)\right]\cos\left(\frac{2\pi a}{\lambda}\right) \tag{4.2}$$

where kernel center is at the origin

$$x = y = 0 \tag{4.3}$$

$$a = x\cos\theta + y\sin\theta \tag{4.4}$$

$$b = -x\sin\theta + y\cos\theta \tag{4.5}$$

Gabor filter wavelength is determined by the equation (4.6) where **w** is the width of the image. Moreover the kernel size **k** is determined by the formulation given in equation (4.7).

$$\lambda = 2^{\lfloor \log_2(w) \rfloor - 5} \tag{4.6}$$

$$k = \left\lfloor \frac{10\lambda}{\pi} \right\rfloor \tag{4.7}$$

The actual convolution kernel is obtained by subtracting mean (DC) value of $\hat{g}$ from itself as in equations (4.8) and (4.9).

$$g_{odd} = \hat{g}_{odd} - \left(\overline{\hat{g}_{odd}} \right) \tag{4.8}$$

$$g_{even} = \hat{g}_{even} - \left(\overline{\hat{g}_{even}} \right) \tag{4.9}$$

In order to extract texture properties with edge and roof elements at a pixel location $I(x, y)$ of road image, standard complex response of Gabor filter, which is given in equation (4.10), is used.

$$\mathrm{I}_{complex}(x, y) = \left(g_{odd} * \mathrm{I}\right)(x, y)^2 + \left(g_{even} * \mathrm{I}\right)(x, y)^2 \tag{4.10}$$

The complex response given by equation (4.10) is calculated for evenly incremented Gabor filter orientation set. For example, all Gabor filter responses with orientation ranging from 0 degree to 177.5 degrees with a 2.5 degrees increment are calculated and form the Gabor filter orientation set. The orientation increment, therefore, the number of orientation (**n**) of Gabor filters are adjusted due to experiments and the ideal orientation number will be discussed in section 4.3.4.

The dominant orientation $\theta(x, y)$ for a pixel location is determined by examining the response energy values of Gabor filter orientation set. The dominant orientation is the orientation whose response energy has the greatest value among the others. By this routine, for all pixels in the entire image the dominant orientations are calculated and dominant orientation matrix is formed. Example dominant orientation matrices calculated for a structured and an unstructured road images through the routine developed in Matlab is depicted in Figure 4-4 and Figure 4-5.

After the dominant orientation matrix of the image computed, a voting procedure is used in order to compute the location of vanishing point. This procedure is presented in the following subsection.

4.3.3 Voting Procedure

After the dominant orientation matrix of the image computed, a voting procedure is used for determining the vanishing point. In the voting procedure, every pixel in the image get votes and the higher vote getter becomes the vanishing point.

Let **IP** is an image pixel in the road image and the dominant orientation of **IP** is $\theta(IP)$. The possible vanishing points for this **IP** should be above **IP** and located on the line defined by L(**IP**, $\theta(IP)$).

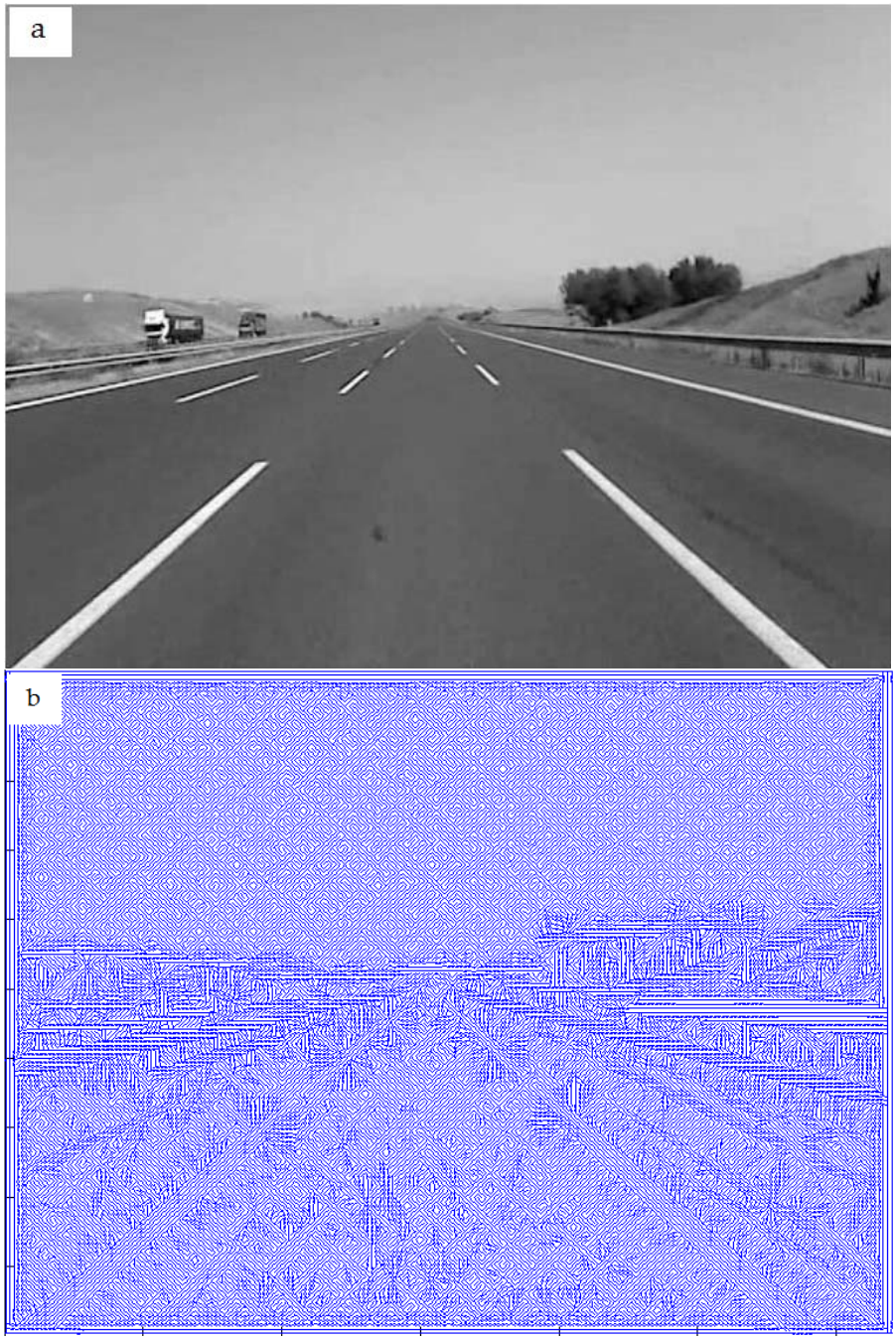

Figure 4-4: (a) A structured road image (b) Its computed dominant orientation matrix

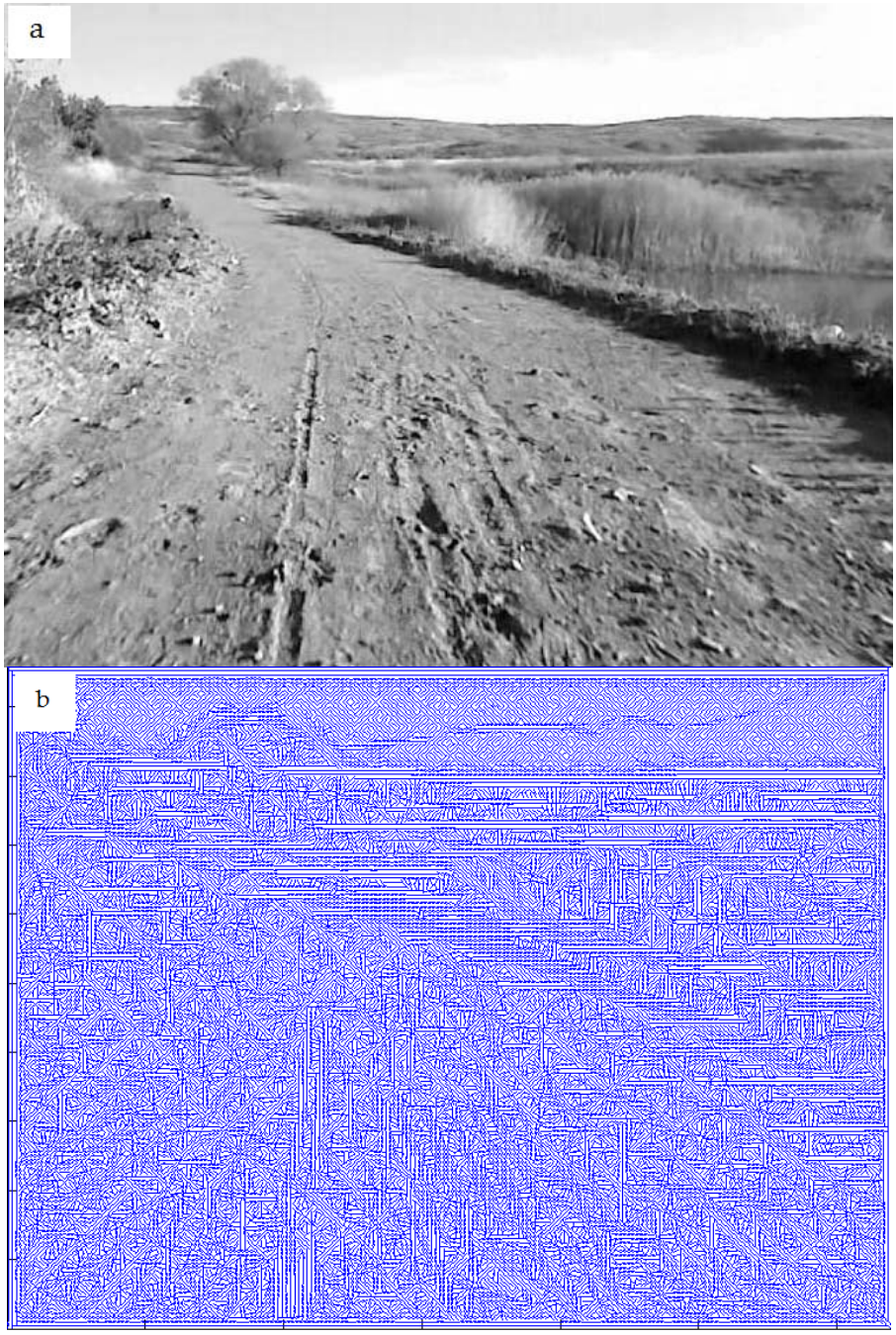

Figure 4-5: (a) An unstructured road image (b) Its computed dominant orientation matrix

While generating dominant orientation matrix, there are **n** Gabor filter orientation. This means that we have an angular resolution of π/n. Therefore, for a vanishing point candidate **VC** being on a line L($\mathbf{IP}, \theta(\mathbf{IP})$) means that the angle $\beta(\mathbf{IP}, \mathbf{VC})$ between **IP** and **VC** should be less than $\pi/2n$. If the vanishing point candidate is on line, that candidates get one vote. Therefore, the voting objective function turns out to be

$$vote(\mathbf{IP}, \mathbf{VC}) = \begin{cases} 1, & \left|\beta(\mathbf{IP}, \mathbf{VC}) - \theta(\mathbf{IP})\right| \le \dfrac{\pi}{2n} \\ 0, & \left|\beta(\mathbf{IP}, \mathbf{VC}) - \theta(\mathbf{IP})\right| > \dfrac{\pi}{2n} \end{cases} \tag{4.11}$$

For all the pixels in the image this voting objective function is applied and the overall votes for all vanishing point candidates are summed up. This voting procedure can be finalized as follows:

$$overallvote(\mathbf{VP}) = \sum_{\mathbf{IP} \in R(\mathbf{VP})} vote(\mathbf{IP}, \mathbf{VP}) \tag{4.12}$$

$R(\mathbf{VP})$ in equation (4.12) means the region of voting for **VP**. In other words, it is the combination of lines on which **VP** is lying. After the application of equation (4.12), one obtains an overall vote matrix which contains the overall voting results for each pixel in the image. The pixel in this matrix with the maximum amount of vote is defined as the vanishing point.

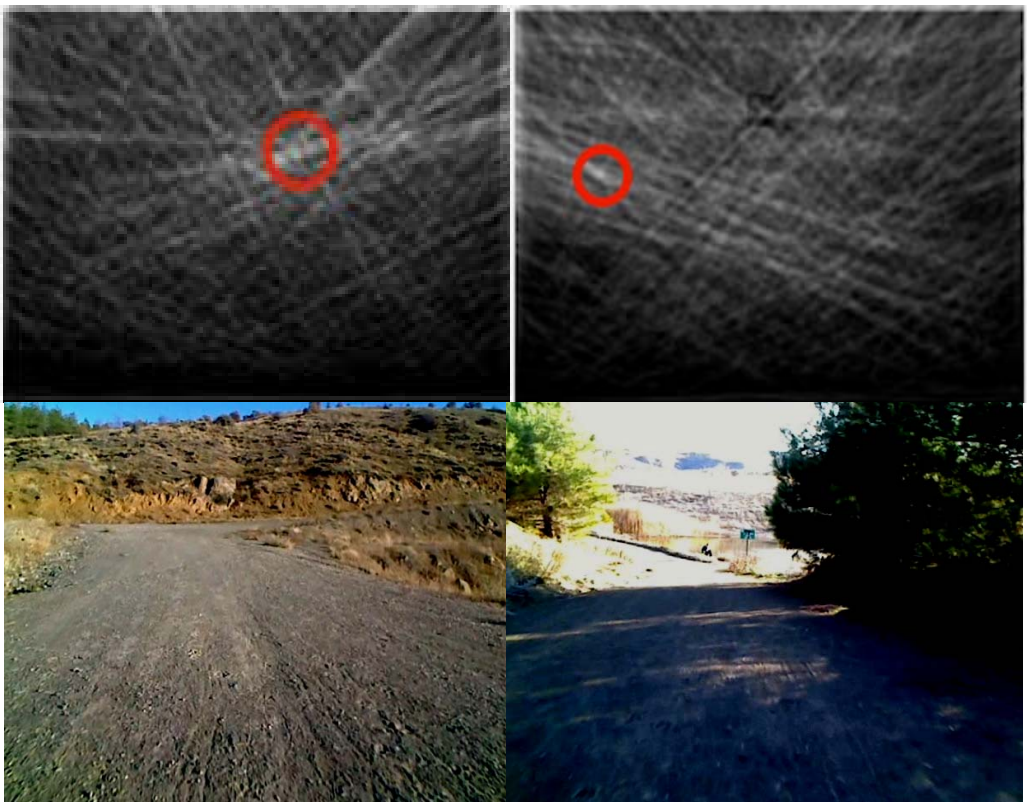

Figure 4-6: Voting matrices for two unstructured road images

In the implementation of the procedure in Matlab environment, to begin with, all images are down sampled 2 levels using a Gaussian pyramid method for reducing the computational cost. Therefore, instead of processing a 640 x 480 pixels sized image, 160 x 120 pixel sized images are processed in the voting procedure. For each pixel on the image a line matrix starting from the related pixel and directed towards the dominant orientation of that pixel is generated. This line matrix has values of 1's in the pixel locations through which the specified line should pass. Then all the line matrices are summed up to obtain the overall vote matrix. An illustration of overall vote matrix generated for two unstructured roads can

be seen in Figure 4-6 with the possible highest vote getter region marked with a red circle.

4.3.4 Experiments & Results

The road videos recorded beforehand (see section 3.6) are used in the experiments which aim to evaluate the performance of vanishing point detection algorithm. For experimentation, 24 video sections each containing 300 frames are cut with different illumination and road texture characteristics from each of the 3 videos resulting in 24 videos. Out of each 24 video, 30 frames were randomly selected, producing 720 frames for training purposes. Among these randomly selected set, %80 of the road images with very similar road curvature and road texture characteristics are eliminated. Finally 144 road images with different characteristics are obtained for experiments. 20 people, 2 of which are professors who are the heads of our research group, SPARC, and 10 of which are my colleagues (M.Sc. Candidates), 6 of which are undergraduate students at Mechanical Engineering Department of METU and 2 of which are non-academic persons labeled the vanishing point on these 144 road images.

Cartesian distance of the experimentally found vanishing point for each image to the average of labeled vanishing points will be computed and a distance histogram will be generated. Resolution of the distance histogram is 10 pixels. This measure is called as **Measure-A**.

Moreover, using the data provided by human labeled vanishing point data, for each image a 2D Gaussian fitting is plotted. Then experimentally found

vanishing point is compared with these 2D Gaussian fittings. This performance evaluating the vanishing point estimation algorithm is called Measure B. The results can be classified in three groups for **Measure-B**. First group is the group in which all the calculated vanishing points stay inside or on the 2D Gaussian fitting curve. Second group is the group where the calculated vanishing points are outside the fitting curve but the maximum distance to the fitting is 40 pixels. The third group is the one where the calculated vanishing points are outside the fitting area and very far from it (over than 40 pixels). Vanishing point experiments are held for these 144 images with different angular resolutions (see section 4.3.3). The results considering measure A and measure B are tabulated for results obtained through all these angular resolutions. Angular resolutions for experiments are selected as 0.01^0 (very fine), 0.5^0 (fine), 1^0 (average), 2^0 (coarse) and 5^0 (very coarse).

First of all, results using Measure-A are depicted in Figure 4-7 to Figure 4-11 for all of the angular resolutions.

It should be noted that the vanishing point algorithm obtain results according to 160 x 120 pixel resolution images. The maximum possible distance in this scheme is 200 pixels, which is the length of the diagonal of the image. Therefore, distances should be evaluated accordingly.

If the distance histograms are examined, it can be easily seen that, with decreasing angular resolution, the accuracy of the vanishing point detection algorithm deteriorates. Computing vanishing points within the region

having a 40-pixel radius and centered at truth location gives vital information about the real location of the vanishing point.

Out of 144 test images, in 121 images vanishing point is computed within that region for angular resolution of 0.01^{o}. For angular resolution of 0.5^{o}, this number is 112. For angular resolution of 1^{o}, vanishing point is found in that acceptable region is 116. For coarser angular resolutions 2^{o} and 5^{o}, these numbers are 90 and 56.

For finest 3 angular resolutions, ratios of acceptable results are 84%, 78% and 81%, respectively. For coarser angular resolutions, this ratios fall down to 63% and 38%. Increasing the angular resolution increases the computational complexity and computation time. Regarding this cost, angular resolution of 1^{o} should be selected as the best performer for Measure-A.

If the images corresponding to the outliers in the distance histograms are examined, it is seen that roads in all of these images either have a high curvature property or contain continuous parallel shadows. Continuous parallel shadows on the road mislead the algorithm to find vanishing point at the right side or at the left side of the image where the end points of the continuous shadows are located.

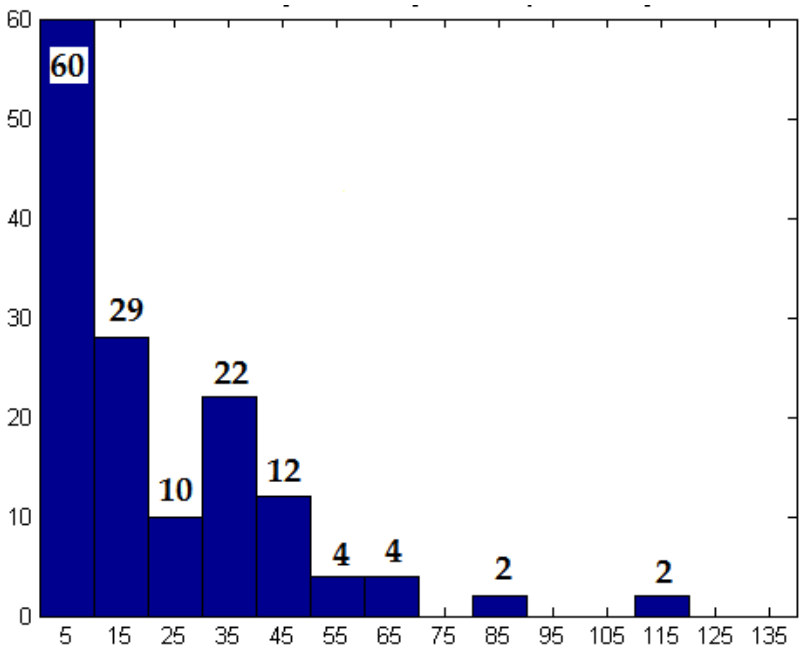

Figure 4-7: Distance histograms for angular res. = 0.01° (Measure A)

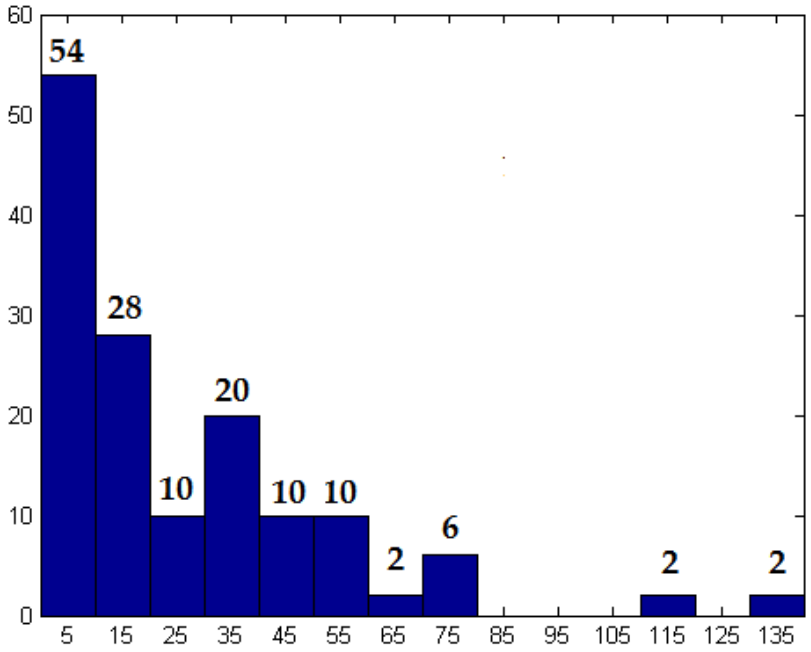

Figure 4-8: Distance histograms for angular res = 0.5° (Measure A)

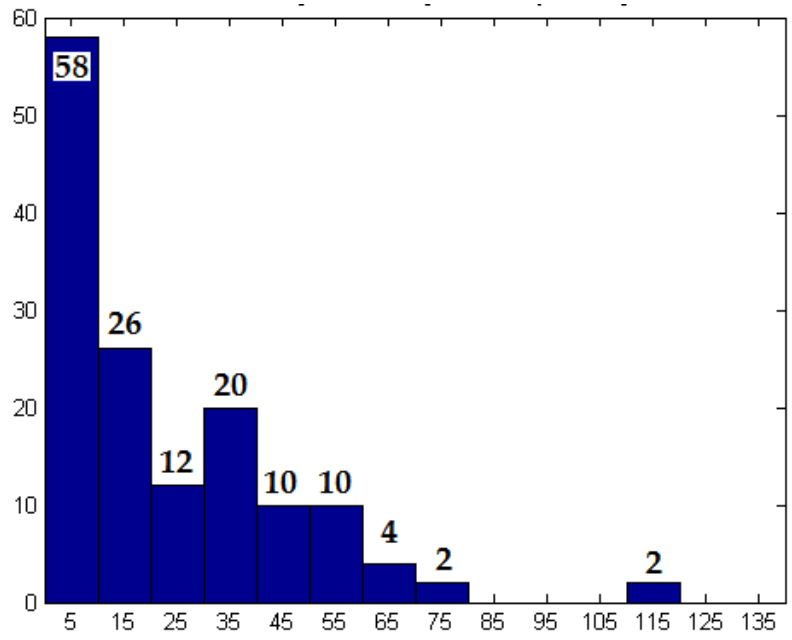

Figure 4-9: Distance histograms for angular res = 1° (Measure A)

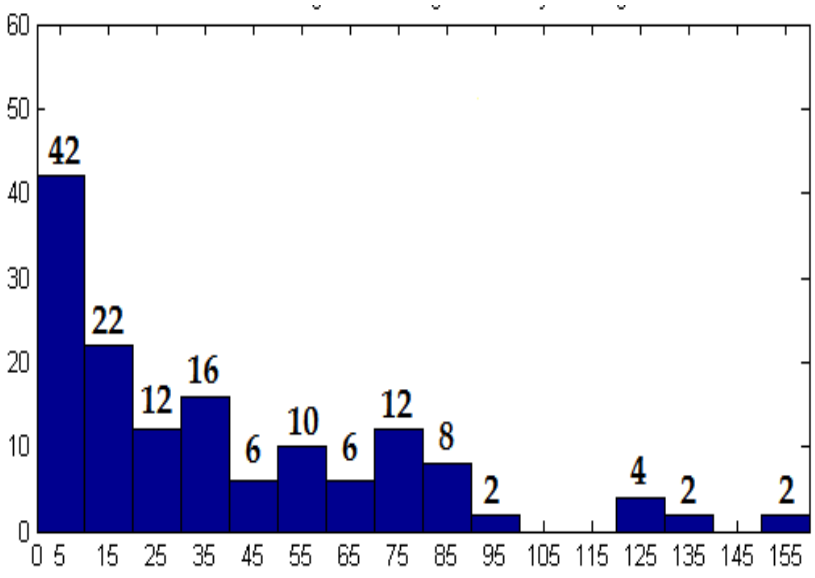

Figure 4-10: Distance histograms for angular res. = 2° (Measure A)

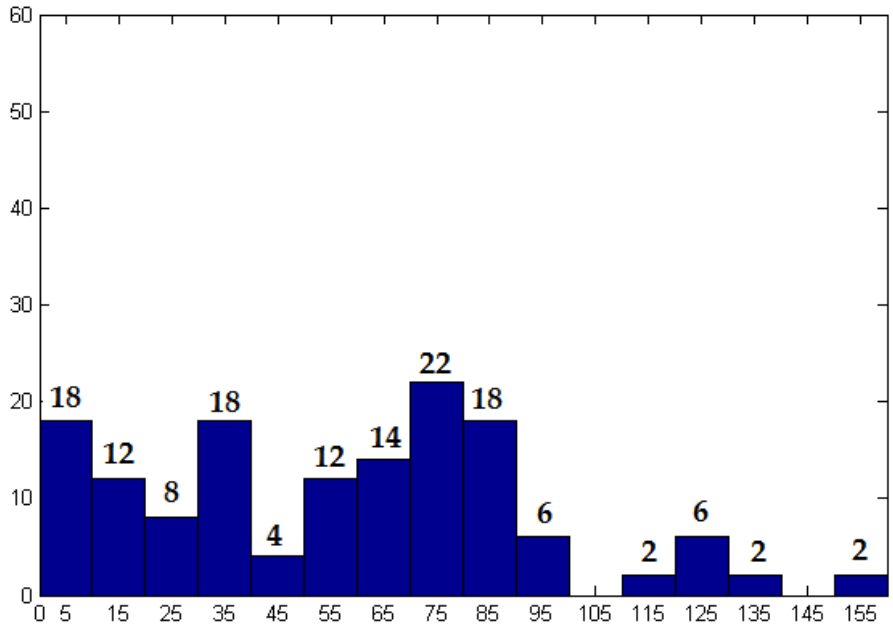

Figure 4-11: Distance histograms for angular res. = 5° (Measure A)

Secondly, results obtained using Measure B will be examined. Sample vanishing point detection images for all three groups of Measure B are depicted in Figure 4-12 to Figure 4-14. Results for Measure B are tabulated in Table 4-1 to Table 4-5 for all of the angular resolutions.

If the result tables are examined, it can be easily seen that, with decreasing angular resolution, the number of the accurately detected vanishing points are decreased. Computed vanishing points belonging to group 1 and group 2 give enough information about the real location of the vanishing point; therefore, points belonging to these groups are accepted as successful.

Figure 4-12: Sample result for Measure B – Group 1

Figure 4-13: Sample result for Measure B – Group 2

Figure 4-14: Sample result for Measure B – Group 3

Table 4-1: Results for angular res. = 0.01° (Measure B)

Group	# of members	Average distance to truth mean (pixels)
1	49	2.188
2	72	9.834
3	23	67.061

Table 4-2: Results for angular res. = 0.5° (Measure B)

Group	# of members	Average distance to truth mean (pixels)
1	45	2.006
2	72	11.562
3	27	69.087

Table 4-3: Results for angular res. = 1° (Measure B)

Group	# of members	Average distance to truth mean (pixels)
1	43	1.988
2	74	12.773
3	34	70.439

Table 4-4: Results for angular res. = 2° (Measure B)

Group	# of members	Average distance to truth mean (pixels)
1	41	1.953
2	51	14.742
3	52	78.235

Table 4-5: Results for angular res. = 5° (Measure B)

Group	# of members	Average distance to truth mean (pixels)
1	34	1.905
2	37	14.946
3	73	80.397

Out of 144 test images, in 121 images vanishing point is computed within that region for angular resolution of 0.01°. For angular resolution of 0.5°, this number is 117. For angular resolution of 1°, vanishing point is found in that acceptable region is 110. For coarser angular resolutions 2° and 5°, these numbers are 92 and 71.

In addition to that, it is seen that average distance to truth vanishing point locations for Group 1 decreases with decreasing angular resolution. In the first look, this phenomenon may seem weird. Since while the angular resolution is decreased, it is expected that average distances should increase. However, with the decreasing angular resolution, the number of computed vanishing points falling within the 2D Gaussian truth curve, which is a very tiny area, decreases. Therefore, the vanishing points staying within the 2D curve are the vanishing points that are computed successfully and that are not affected from the angular resolution. Therefore, it is logical for average distances to decrease with decreasing angular resolution.

Whereas, average distance to truth vanishing point locations for Group 2 and Group 3 increases with decreasing angular resolution. This is expected since the coarser the resolution is the poorer the vanishing results are. Computed vanishing points, which are outside the 2D Gaussian truth curve, become farther when the angular resolution is decreased. Therefore, it is logical for average distances to increase.

For finest 3 angular resolutions, ratios of successful results are 84%, 81% and 79%, respectively. For coarser angular resolutions, this ratios fall down to 63% and 49%. It should be noted that, increasing the angular resolution increases the computational complexity and computation time. Regarding this cost, angular resolution of 1^{o} should be selected as the best performer for Measure B.

Again if the images falling in the third group (poor performers) are examined, it is seen that images contains roads degraded with continuous parallel shadows, which mislead the algorithm to find vanishing point the sides of the image. Moreover, in images containing sharp turning roads, the algorithm performance deteriorates.

4.3.5 Conclusion

In this subchapter, an algorithm for vanishing point estimation depending on the road texture characteristics was presented. The results assure that the vanishing point estimation algorithm works satisfactorily for variety of road profiles, expect the road with high curvature characteristics. The position of vanishing point is critical for steering purposes since it gives an idea to the robot about toward which direction the road leans. Therefore, this information will be used in order to compute the reference heading of the robot, which is presented in CHAPTER 5.

It is seen that with the road with a high curvature characteristics, the vanishing point estimation performance become poor. But for straight roads and roads with low curvature characteristics, vanishing point estimation algorithm work pretty good. Moreover, in roads degraded by continuous parallel shadows, algorithm performance is comparably poor.

In addition to that, it is found that among the angular resolutions of 0.01^0 (very fine), 0.5^0 (fine), 1^0 (average), 2^0 (coarse) and 5^0 (very coarse), the optimum angular resolution is 1^0. Since the performance of the algorithm this resolution is very similar to that of algorithms using 0.01^0 and 0.5^0.

Moreover, another reason for selecting angular resolution as 1^0, its computational load of is much less than that of 0.01^0 and 0.5^0.

A modification should be done to the voting procedure, in order to achieve more accurate estimation of vanishing points belonging to high curvature roads. This modification can be the extraction of multiple high vote getters from the voting matrix instead of the extraction of the highest vote getter. Then one can evaluate them to estimate the correct vanishing point. Since it is observed that, the real vanishing point appears to be the one in the highest 10 vote getter in most of the poorly detected vanishing point cases. However, this will remain as a future work to this academic research.

4.4 Color Based Unstructured Road Detection Algorithm

4.4.1 Introduction

Unstructured roads contain various irregular properties, perception and tracking of which, are not easy via image processing. Some unstructured roads are created on purpose; some just emerge as a result of continuous use, and some roads still out there to be discovered. These roads have no deliberately-made markings or features, and their properties change very drastically from one setting to another. Detecting drivable surfaces in unstructured settings by using limited features is not an easy problem to tackle. Hence, unstructured road detection for autonomous vehicles is a research area open for significant contribution.

The most important issue in ANN training is the selection of related and distinctive feature vectors. In this section, a method utilizing feature vectors built up using color information of road images. Color information may be based on Red-Blue-Green (RGB) color space or on Hue-Color-Saturation (HSV) Color space or on both of them. In Figure 4-15, all color channels of RGB and HSV color spaces of an unstructured road image are illustrated.

Most of the road detection routines available in the literature (see Section 2.2.2 and 2.3.2) utilize only RGB color space in their solution. However, HSV color space has distinctive characteristics. One of these properties of HSV color space is that it is more insensitive to shadow and other illumination artifacts. Shadows can easily be distinguished in saturation channel since parts of image with shadow shines white in saturation channel image (Figure 4-15.a)

Moreover, in some situations, the drivable region observed in HSV color space (especially images obtained from saturation channel of HSV color space) is more obvious than its counterparts obtained by RGB color space (Figure 4-15.b).

In the following subsections, an overview of the method introducing the basic concepts related to it, feature generation descriptions, utilization of the color spaces in order to extract feature vectors will be discussed.

4.4.2 Overview of the Method

The method proposed in this section based on an ANN structure fed with different input and output feature vectors, extraction methods of which will be described throughout this chapter.

The ANN used in this algorithm consists of a one hidden layer with a specified number of hidden neurons. The weights of the ANN are updated using back-propagation method using "unipolar sigmoid" activation function.

The number of the hidden neurons is selected as the nearest integer mean of the number of inputs and outputs. For example, let the number of inputs is 60 and the number of outputs is 121, the number of hidden neurons in the hidden layer is selected as 91.

In the proposed method, 'divide and conquer' principle is adopted and the road image, which has a 640 x 480 pixels resolution, is divided into smaller kernels, for which the feature vectors are computed. In other words, each feature vector, which is to be fed to the ANN, is computed for a smaller area of image called kernels.

Kernels are the square regions which is a part of the road image. Independent of their sizes, centers of kernels are apart from each other with 10-pixel intervals vertically and horizontally. Restating, there is a kernel region centered at every 10 pixel of image horizontally and vertically. Kernel size should an odd number assuring the symmetry.

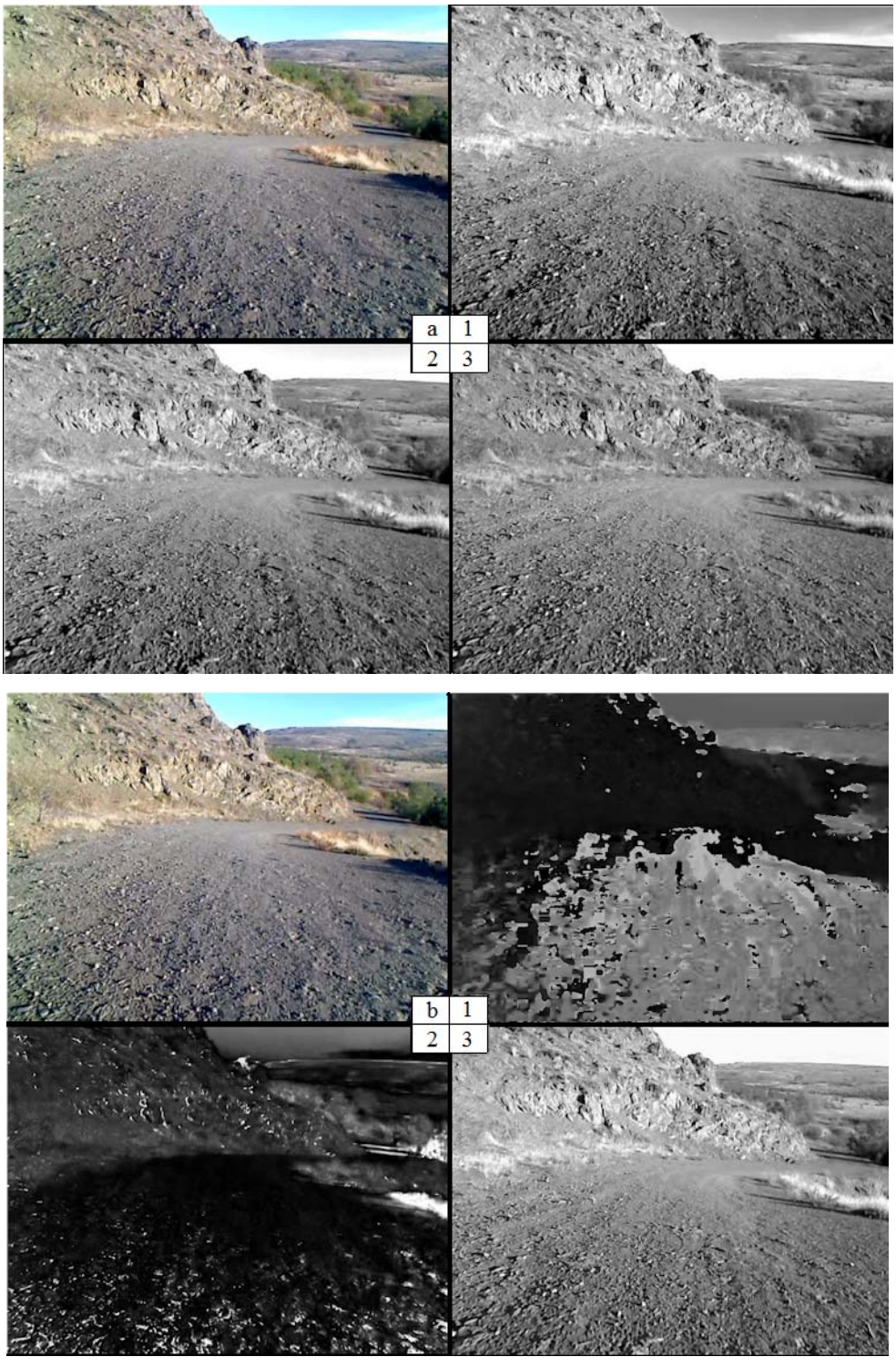

Figure 4-15: (a) Original road image – (1) Red / (2) Green / (3) Blue channel views of a sample road image. (b) Original road image – (1) Hue / (2) Saturation / (3) Value channel views of a sample road image.

To ensure that the filter kernels remained entirely in the image, a 20-pixel margin is left (This safety margin can be modified according to the kernel size). An example division of the road image, which has a 640 x 480 pixels resolution, is given in Figure 4-16. The red dots represent the center locations of the kernels. The outside of the black rectangle represents the safety band.

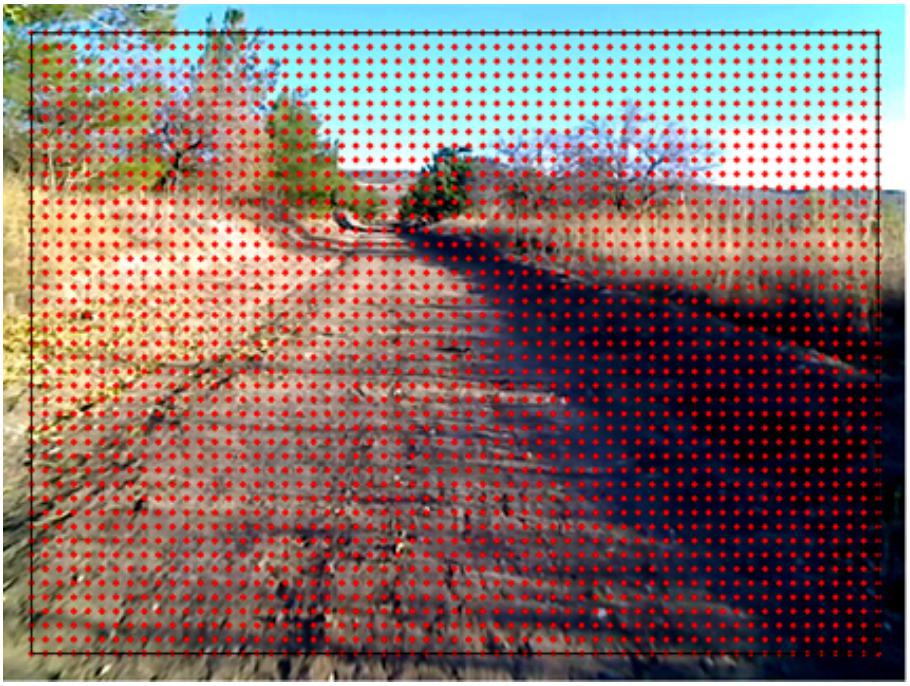

Figure 4-16: Centers of the kernels

For a 640x480 pixels resolution image, the partition of image proposed above results in 2745 kernels for each image. By changing the interval between the areas for which the feature vectors are computed and the safety region in the exterior region of image, number of the kernels can be

changed. However throughout this chapter the safety band is taken as 20 pixels and the interval between kernel centers is taken as 10 pixels. However the kernel size will be changed starting from 11 x 11 pixels since 11 x 11 kernel size is the minimum kernel size which a human can classify the color features. The other kernel sizes that will be used are 15, 31 and 41.

For a specified kernel size, interval of kernel center and safety band parameters both input and output feature vectors are computed and fed into the generated ANN. In the next 2 subsections it is described how these input and output feature vectors are generated.

4.4.3 Input Feature Vector Extraction

After the parameters related to the kernel size are selected, input feature vectors for these kernels should be computed. There are several input feature vector schemes using color spaces. Most of them utilize the color histograms. In the following subsections, feature vector types are described in detail. These are independent histogram based, joint histogram based, combined histogram based features and features using all raw information included in the kernel. Tutorial on how to extract input feature vectors using the codes included in the CD enclosed by this thesis can be found at Appendix C.

In all of the input feature vectors, the location (x,y) of the kernel is included. The purpose for including the kernel location is to speed up the training phase. Since the locations point ANN that the road regions are in the lower part of the image and this helps ANN to converge in ambiguous situations.

4.4.3.1 Independent Histogram Based Features

RGB color space is composed of three channels, namely Red, Green and Blue. Each channel has an intensity value ranging from 1 to 256. In other words, there are three intensity values (red intensity, blue intensity and green intensity) characterizes a pixel using RGB representation.

Histogram of channel information of an image includes information about the number of pixels which possess a particular intensity value. For example, there are 25 pixels possessing an intensity value of 255, histogram value of the 255^{th} intensity value of that channel is 25. In Figure 4-17.a, regular histogram of the red channel of a road image is given.

Similarly, histogram of HSV color space channels can also be computed. HSV color space is composed of three channels, namely hue, saturation and value. Hue channel has the color information, values ranging from 0^{o} to 360^{o}. Every degree value in the channel corresponds to a color value. For example, both 0^{o} to 360^{o} correspond to a tone of red value. Saturation channel ranging from 0 to 1.0 given the saturation information of colors and finally value component whose values are ranging from 0 to 1.0 gives the brightness information. Since hue channel is a periodic channel, it is meaningless to generate a histogram for it, while for saturation and value channels n-bin histograms can be built. After the bin selection of the histogram is performed, feature vector can be built up.

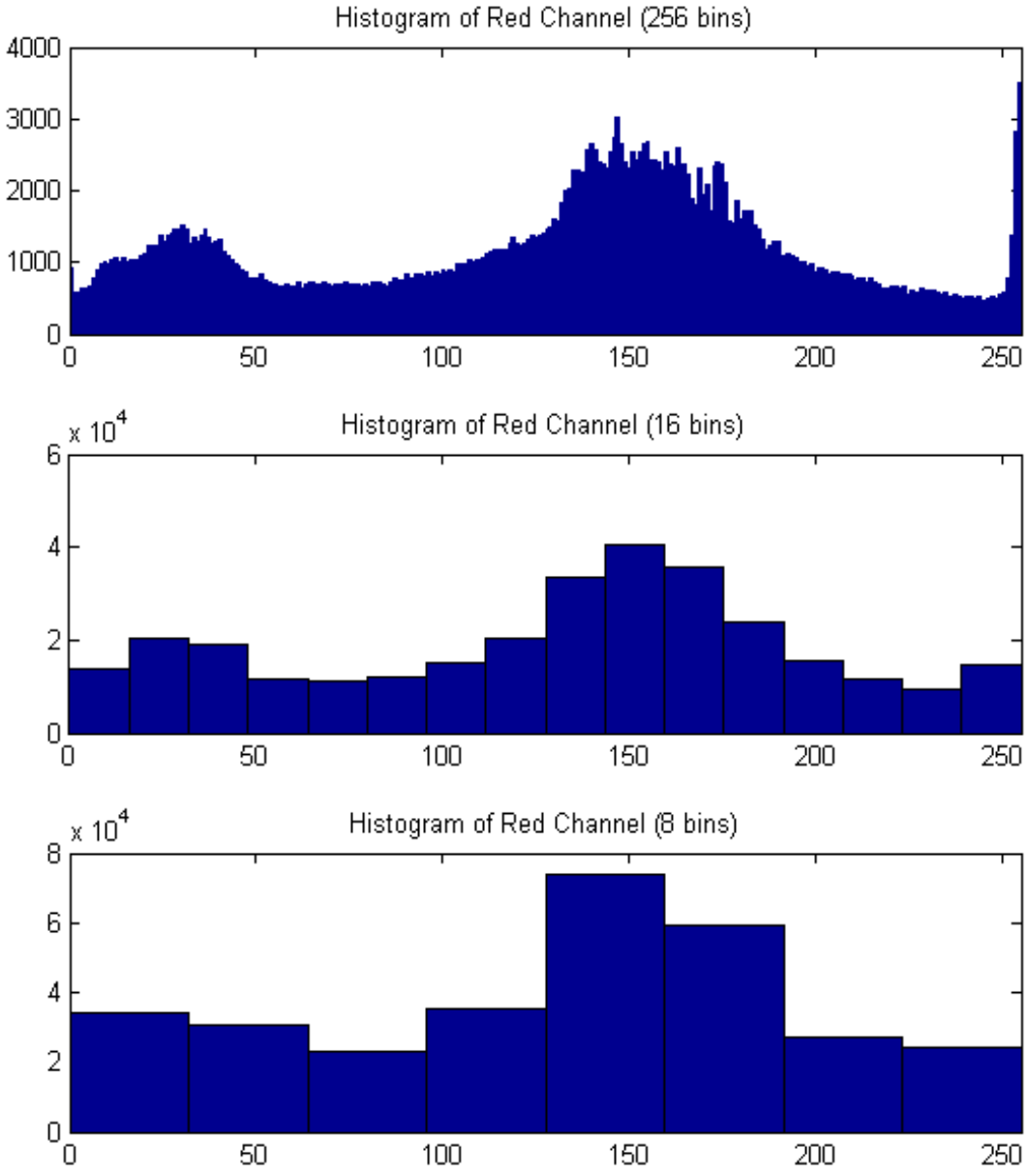

Figure 4-17: (a) Regular histogram (b) 16-bin hist. (c) 8-bin hist.

The feature vector utilizing independent histograms can be explained easily by giving a simple example. Let our kernel size be 11. This means that we extract feature vectors for image patches with a size of 11 x 11 pixels. Assume only RGB channels are used for feature and our kernel is located at $(x, y) = (116, 316)$ of 640 x 480 pixels sized road image. Let we construct a 4-bin histograms for each channel for this example. Therefore, we should expect a 14 x 1 sized feature vector. An example image patch is extracted from a road image and the feature extraction steps are shown in Figure 4-18. Note the normalization taking place during the feature extraction.

First of all, 4-bin independent histograms for each channel are extracted. Then the histograms are normalized by dividing results with the number of pixels within the selected kernel (k^2 = 121 for example case). The center location of kernel is also normalized. Abscise of the center location is normalized by the width (w = 640 for example case) and ordinate of the center is normalized by height (h = 480 for example case) of the original road image. In final step, all these vectors are combined to build the input feature vector for the specified kernel.

Every feature vector in this subchapter includes histogram information within each kernel and the center location of the kernel. In other words, the feature vector utilizing m-color channels with n-bin independent histograms has a size of $(m \cdot n + 2) \times 1$.

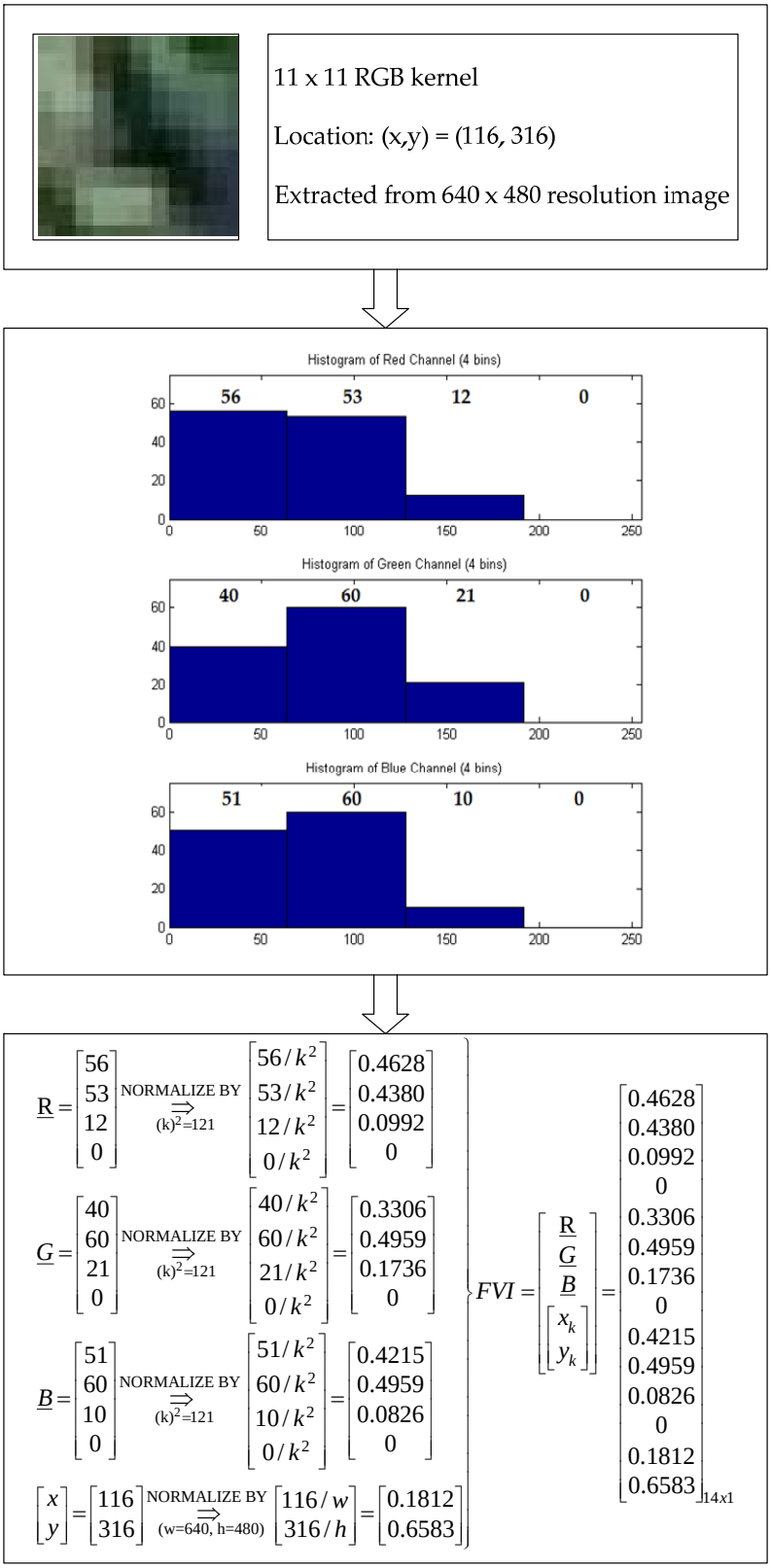

Figure 4-18: RGB 4-bin independent histogram feature vector

4.4.3.2 Joint Histogram Based Features

Second color feature is joint color histogram. This time, while extracting the histogram information out of the color channels, the color channels are not independent from each other. Similarly, the intensity values of the channels are divided into bins but in this case these bins are related to each other. The number of dimensions of the joint histogram is related to the number of channels of which the joint histogram is extracted. For example a joint histogram utilizing color information of three color channels has 3 dimensions and it can be depicted as a cube.

In order to understand the joint histogram concept, starting with a 2-dimensional example would be more appropriate. Let our kernel size be 11. This means that we extract feature vectors for image patches with a size of 11 x 11 pixels. We will use only Red and Blue color channels for our feature vector and our kernel is located at $(x, y) = (116, 316)$ of 640 x 480 pixels sized road image.

Let us construct a 2D plot whose x-axis represents the intensity values of Red color channel and y-axis represents the intensity values of Blue color channel. In the graph, all 121 pixels of the kernel are distributed according to their red and blue color channel intensity value information Figure 4-19.

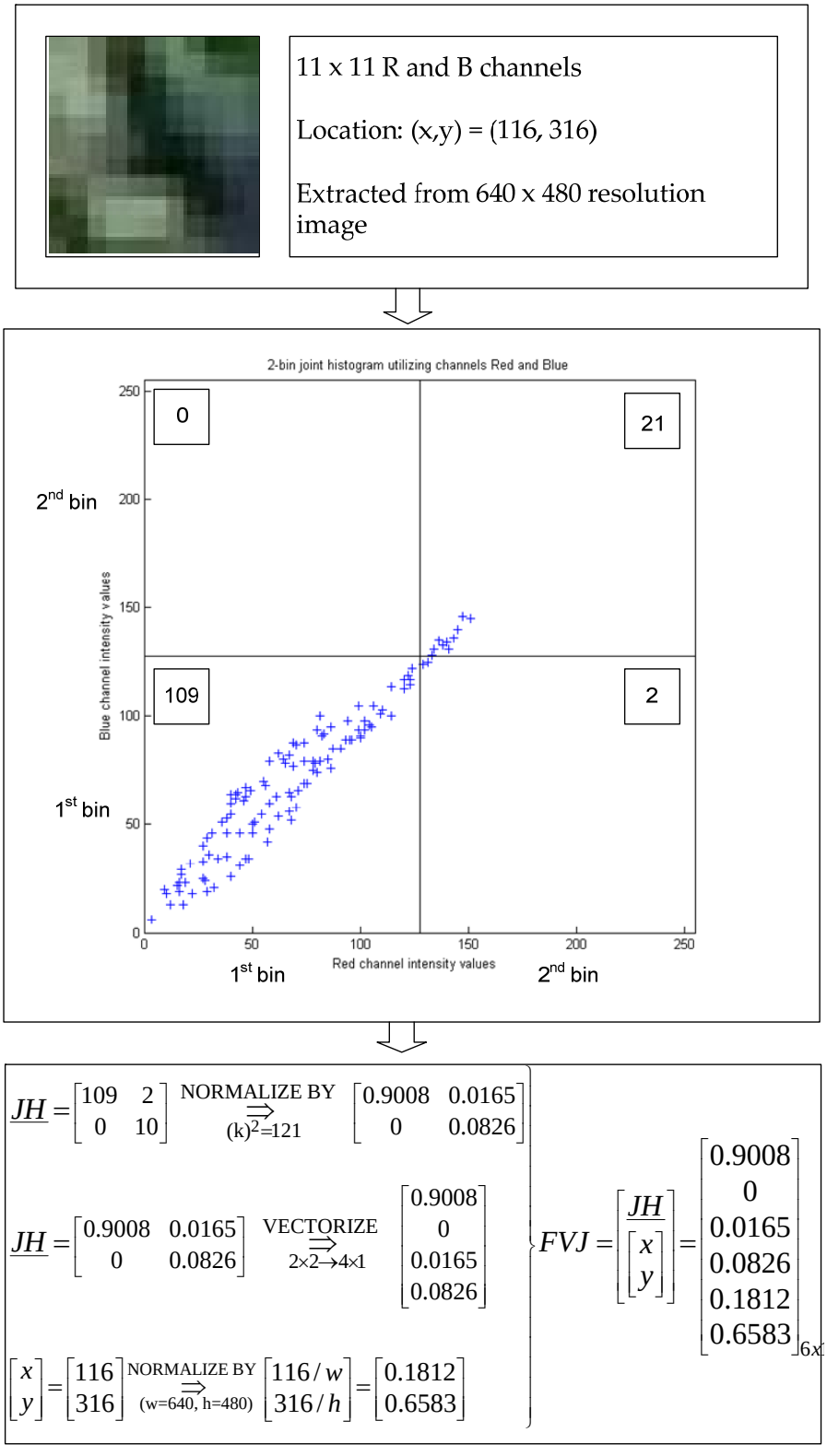

Figure 4-19: Red and blue channels 2-bin joint histogram feature vector

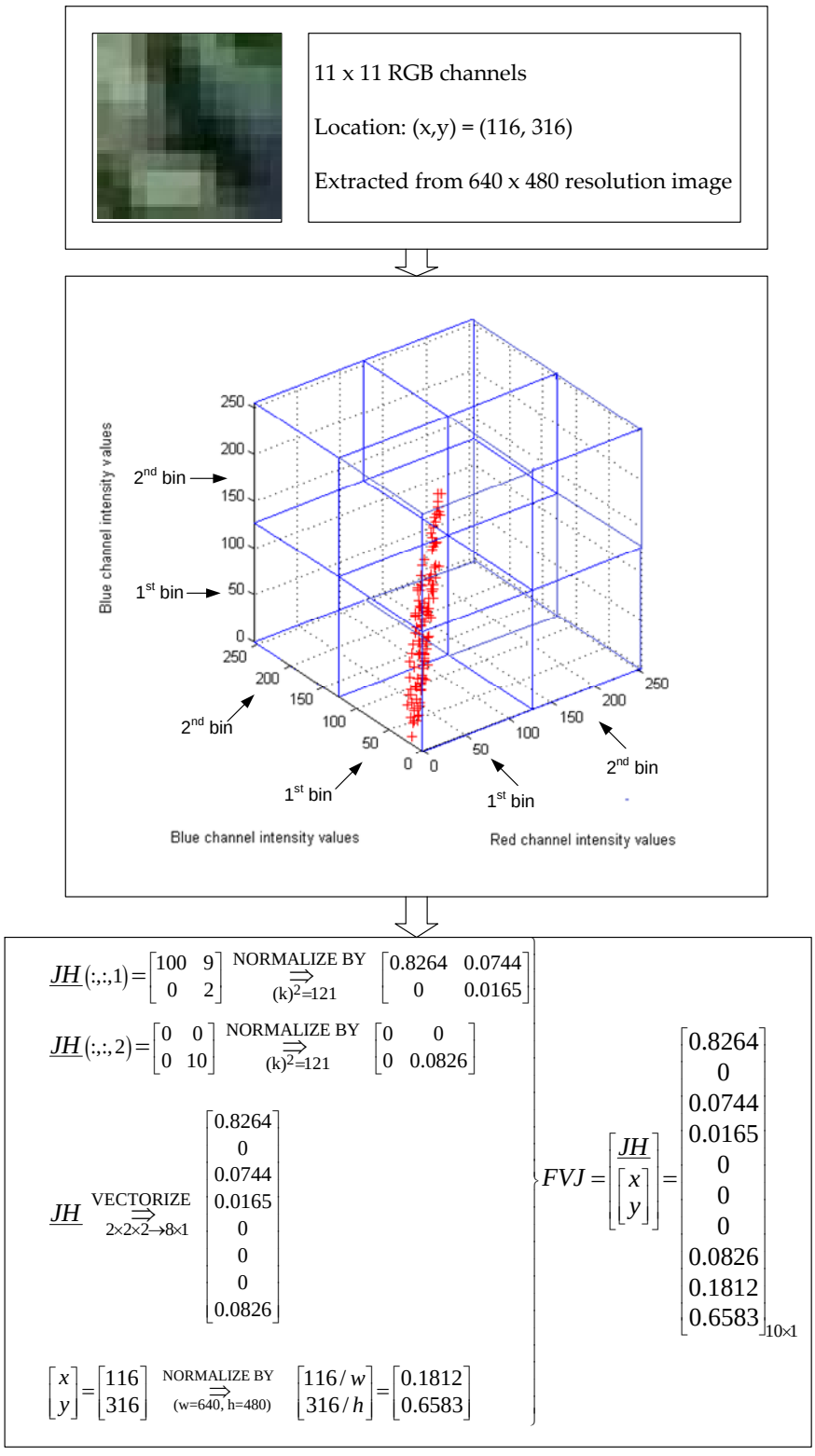

Figure 4-20: RGB channels 2-bin joint histogram feature vector

After joint histogram of channels are obtained, 2D joint histogram is normalized by dividing each cell with the total pixel number within the kernel ($k^2 = 121$ for example case). After the normalization, it is combined with the normalized center locations of the kernel to build the final feature vector.

Joint histogram extraction of all of the three RGB channels is very similar. The procedure of extraction of feature vector utilizing 3D joint histogram and its representation are depicted in Figure 4-20.

Every feature vector in this subchapter includes joint histogram information within each kernel and the center location of the kernel. In other words, the feature vector utilizing m-color channels with n-bin joint histograms has a size of $\left(m^n + 2\right) \times 1$.

4.4.3.3 Combined Histogram Based Features

Third color feature is the combination of independent color histograms and joint color histogram. The feature vector utilizing joint histogram is added at the end of feature vector using independent histogram information. Finally the normalized center location is also added at the end of this combination.

For example, feature vector using 3 RGB channel 16-bin independent histogram information (FVI) and 3 RGB channel 4-bin joint histogram (FVJ) information will be as in equation (4.13).

$$\left.\begin{aligned}\underline{FVI} &= \begin{bmatrix}\underline{R}_{16\times1}\\ \underline{G}_{16\times1}\\ \underline{B}_{16\times1}\end{bmatrix}_{48\times1}\\ \underline{FVJ} &= \underline{JH}_{64\times1}\end{aligned}\right\} \underline{FVC} = \begin{bmatrix}\underline{FVI}\\ \underline{FVJ}\\ \begin{bmatrix}x\\ y\end{bmatrix}\end{bmatrix}_{114\times1} \tag{4.13}$$

Every feature vector utilizing m_j-color channels with n_j-bin joint histograms and m_i-color channels with n_i-bin joint histograms has a size of $\left(m_i \cdot n_i + m_j{}^{n_j} + 2\right)\times 1$.

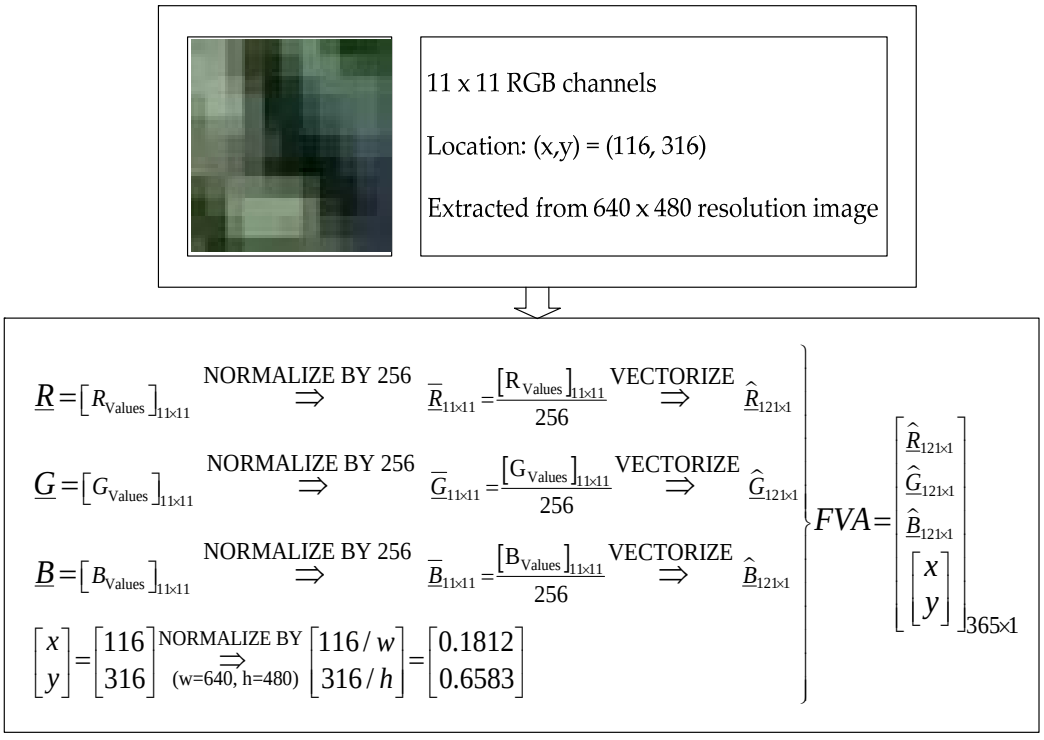

Figure 4-21: RGB channel feature vector utilizing all information

4.4.3.4 Feature Using All Intensity Value Information within the Kernel

Fourth color feature is purely all the intensity value information included within the kernel. For example, for a kernel size k = 11, using all of the RGB channels feature vector using this subchapter's scheme is extracted as in Figure 4-20.

Every feature vector in this subchapter includes purely all intensity value of the color channels within a kernel and the center location of the kernel. In other words, the feature vector utilizing m-color channels with k x k pixels sized kernel has a size of $\left(m \cdot k^2 + 2\right) \times 1$.

4.4.4 Output Feature Vector Extraction

After input feature vectors are generated, for training ANN output feature vectors should be created. Regions in the images extracted from the recorded videos are labeled as road (1) and non-road (0) regions and at the end of the labeling process a binary road – non-road matrix (Truth matrix) is obtained for all images.

After the generation of truth matrix, output feature vectors are extracted for kernels centered at the locations specified in Figure 4-16. The kernel size of output feature vectors is fixed at 11 x 11 pixels. An example of output feature vector extraction for a pseudo kernel of size 5 x 5 is given in Figure 4-22. Tutorial on how to extract output feature vectors using the codes included in the CD enclosed by the thesis can be found at Appendix C.0

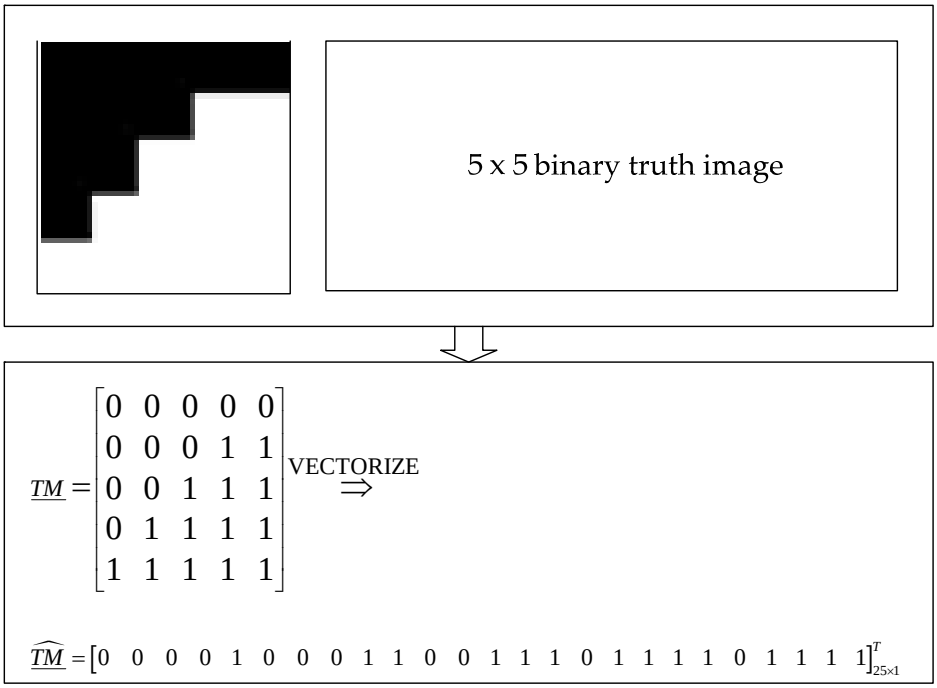

Figure 4-22: Output feature vector (kernel = 11)

4.4.5 Application of the Method

After the extraction of input and output feature vectors, an ANN suits these inputs and outputs are constructed and in the specified order, feature vectors representing all the training images are fed to the ANN, and ANN evolves.

For training, 24 video portions with different road and illumination characteristics are selected. Each video is approximately 300 video frames long. Among 300 frames of each video portion, randomly 30 frames are selected totaling up to 720 frames for training. Road and non-road regions of these 720 road images are labeled by selecting the road area with a mouse for constructing the output side of training data. Then this truth

image is divided keeping the information in kernels separately. For each kernel whose size is 11x11 pixels, there are 121 road/non-road labels.

A sample training case can be described as below. Assume that the input kernels are generated using combined histogram with 4-bin joint histogram $\left(m_j = 4\right)$ and 8-bin independent histogram $\left(m_i = 8\right)$, all of which utilizing all RGB channels $\left(n_j = n_i = 3\right)$ of the image. Let al.l the images have a resolution of 640 x 480 pixels and the kernel size of the feature vectors are 11 x 11 pixels and centered at the locations described in section 4.4.2. In this case, there will be 61 x 45 = 2745 feature vectors extracted from an image with a 640 x 480 pixel resolution.

Using the schemes described in sections 4.4.3 and 4.4.4, feature vectors will be extracted. For input feature vectors for 11 x 11 pixels sized kernel with the specified properties above, one will obtain $\left(8\times 3+4^3+2\right)\times 1=90\times 1$ sized vectors will be obtained. Output feature vectors for each kernel have a size of 121×1 as described in section 4.4.4.

This means that for generated ANN we have 90 input neurons and 121 output neurons. After the number of inputs and outputs became clear, number of the hidden layer neurons should be decided. There are three common conventions used as thumb rules in the literature for specifying the number of hidden layer neurons [86]:

- "The number of hidden neurons should be in the range between the size of the input layer and the size of the output layer. "

- "The number of hidden neurons should be 2/3 of the input layer size, plus the size of the output layer. "
- "The number of hidden neurons should be less than twice the input layer size. "

All these number conventions fall into the region between the size of the input layer and the size of the output layer. For that reason in this study, number of hidden neurons is defined as the nearest integer value of the average of input and output layer sizes. For that example number of hidden neurons is specified as 106. However, one should select hidden neuron numbers step by step updating it and then evaluating the performance of the ANN for each case.

After these parameters are set, the ANN is generated using *nprtool* ANN training for pattern recognition toolbox of MATLAB. After that, the convergence parameters should be set for training. Convergence parameters are parameters that tell the ANN where to stop training. The default parameters for ANN training performed in this study can be seen in Table 4-6.

Table 4-6: Convergence parameters for ANN training

Max. # of epochs	1000
Min. error stop	1e-06
Max. validation check	30
Min error	1.2e-02

Tutorial on how to train an ANN using the routines included in the CD enclosed by this thesis can be found at Appendix C

4.4.6 Experiments & Results

After the ANNs were trained, input feature vectors should be generated for a new road image set in order to evaluate the performance of the algorithm. 720 test images, which are other than the training set, are selected from the available video recordings.

After the selection of 720 test images, corresponding input feature vectors are extracted for all of test images and these feature vectors are fed into corresponding trained ANNs. The outputs of the ANNs are arranged and the binary road/non-road image is obtained.

In this study, 33 different type input feature vector is generated and for each type an ANN is trained. These input feature vectors can be classified into three main groups given in Table 4-7,

Table 4-8 and Table 4-9. First group composed of input feature vectors utilizing all of the intensity values within the corresponding kernel. Second group consists of input feature vectors using independent histograms. Input feature vectors within third group make use of the combined histogram concept.

The types of these input feature vectors and corresponding hidden neuron numbers are given in Table 4-10. All the feature vector sizes are computed according to the routine given in sections 4.4.3 and 4.4.4. Corresponding

hidden neuron numbers are calculated according to the convention given in section 4.4.5. Method codes given in Table 4-10 are composed of three main parts (Figure 4-23). First part of the name represents all the color channels used in feature vector. Second part represents the size of the kernel. And the final part represents the histogram type such as independent (e.g. i8), joint (e.g. j4) or both (e.g. i8 j4), followed by the corresponding bin number.

Sample results for 4 different ANNs trained with 4 different input feature vector set are given in Figure 4-24. Regions labeled with green circles represent the road regions found by the methods. Figure has the corresponding input feature types in its caption.

Table 4-7: Input feature vectors utilizing all information within a kernel (RGB = Red/Green/Blue, SAT=Saturation, VAL=Value)

	KERNEL SIZE		
CHANNELS	**11**	**15**	**31**
RGB	ALL	ALL	
SAT	ALL	ALL	
SAT+VAL	ALL	ALL	
RGB+SAT	ALL	ALL	

Table 4-8: Input feature vectors utilizing independent histograms

	KERNEL SIZE		
CHANNELS	**11**	**15**	**31**
RGB	i8	i8	i8
SAT	i8-i16	i8-i16	i8-i16
SAT+VAL	i8	i8	i8
RGB+SAT	i8-i16	i8-i16	i8-i16

Table 4-9: Input feature vectors utilizing combined histograms[3]

	KERNEL SIZE		
CHANNELS	**11**	**15**	**31**
RGB	i8 - i16	i8 - i16	i8 - i16
SAT			
SAT+VAL		i8	
RGB+SAT			

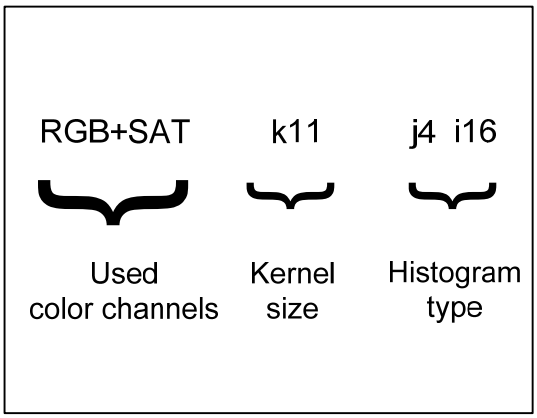

Figure 4-23: Input feature vector naming convention

As it can be seen in the figures, ANN trained by using RGB k31 j4 i8, SAT k15 ALL and RGB+SAT k15 i16 input feature vectors finds nearly all the road regions, while ANN trained by using SAT k11 ALL found some non-road areas as road.

Tutorial on how to train an ANN using the routines included in the CD enclosed by this thesis can be found at Appendix C

[3] All joint histogram included in the combined histogram concept has a 4-bin resolution.

Table 4-10: All types of input feature vectors and corresponding hidden neuron numbers.

METHOD CODE	Size of Input Feature Vector	Size of Output Feature Vector	Number of Hidden Neurons
RGB k11 ALL	365 x 1	121 x 1	243
RGB k15 ALL	667 x 1	121 x 1	394
SAT k11 ALL	123 x 1	121 x 1	122
SAT k15 ALL	227 x 1	121 x 1	174
SAT+VAL k11 ALL	244 x 1	121 x 1	183
SAT+VAL k15 ALL	452 x 1	121 x 1	287
RGB+SAT k11 ALL	486 x 1	121 x 1	304
RGB+SAT k15 ALL	902 x 1	121 x 1	512
RGB k11 i8	26 x 1	121 x 1	74
RGB k15 i8	26 x 1	121 x 1	74
RGB k31 i8	26 x 1	121 x 1	74
SAT k11 i8	10 x 1	121 x 1	66
SAT k15 i8	10 x 1	121 x 1	66
SAT k31 i8	10 x 1	121 x 1	66
SAT k11 i16	18 x 1	121 x 1	70
SAT k15 i16	18 x 1	121 x 1	70
SAT k31 i16	18 x 1	121 x 1	70
SAT+VAL k11 i8	18 x 1	121 x 1	70
SAT+VAL k15 i8	18 x 1	121 x 1	70
SAT+VAL k31 i8	18 x 1	121 x 1	70
RGB+SAT k11 i8	34 x 1	121 x 1	78
RGB+SAT k15 i8	34 x 1	121 x 1	78
RGB+SAT k31 i8	34 x 1	121 x 1	78
RGB+SAT k11 i16	66 x 1	121 x 1	94
RGB+SAT k15 i16	66 x 1	121 x 1	94
RGB+SAT k31 i16	66 x 1	121 x 1	94
RGB k11 j4 i8	90 x 1	121 x 1	106
RGB k15 j4 i8	90 x 1	121 x 1	106
RGB k31 j4 i8	90 x 1	121 x 1	106
RGB k11 j4 i16	114 x 1	121 x 1	118
RGB k15 j4 i16	114 x 1	121 x 1	118
RGB k31 j4 i16	114 x 1	121 x 1	118
SAT+VAL k11 j4 i8	34 x 1	121 x 1	78

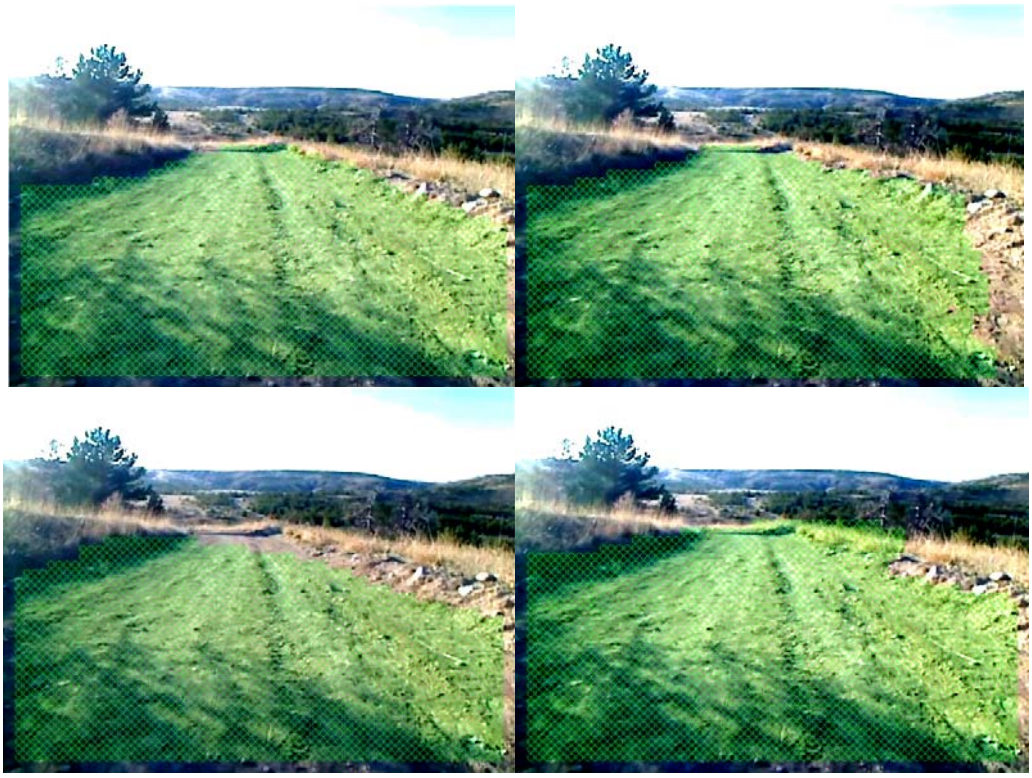

Figure 4-24: ANN results trained by using (a) SAT+VAL k15 i8 (b) SAT k15 ALL (c) RGB+SAT k15 i16 (d)SAT k11 i8 input feature vectors

For comparison, three measures are selected. One of them, **Measure-A**, is the percentage of the correctly detected road points. This measure is defined as the ratio of the number of pixels detected as road by the method and matches with the road points in truth image to the total number of pixels that are labeled as road in truth image. The other one, **Measure-B**, is the percentage of the incorrectly detected road points. This measure is described as the ratio of the number of pixels detected as road by the method but labeled as non-road in truth image to the total number of pixels that are labeled as non-road in truth image.

The final comparison measure is Kappa coefficient (**Measure – C**). Kappa coefficient is used in order to evaluate the level agreement between two voters who vote a two-choice case. In our case, this two-choice case is the road/non-road labels for a given image. One of the voter is the person who label the road regions in the images (ground truth), and the other voter is the algorithm who finds the road regions out of images. The binary labeled images are assumed to be the votes for both voters. In order to calculate the kappa coefficient, first of all a table of agreement should be prepared as in Table 4-11.

Table 4-11: Agreement table

		Algorithm Result		
		ROAD	**NON-ROAD**	**TOTAL**
Ground Truth	**ROAD**	P_{RR}	P_{RN}	$P_{R\_}$
	NON-ROAD	P_{NR}	P_{NN}	$P_{N\_}$
	TOTAL	$P_{\_R}$	$P_{\_N}$	

P_{RR} is the ratio of number of pixels labeled as road in both of the binary vote images to the total number of pixels in vote image. Similarly, P_{NN} is the ratio of number of pixels labeled as non-road in both of the binary vote images to the total number of pixels in vote image. P_{RN} is the ratio of number of pixels labeled as road in ground truth image but labeled as non-road in algorithm's vote image to the total number of pixels in vote image. Finally, P_{NR} has a similar definition to that of P_{RN}.

In order to evaluate Kappa coefficient, observed level of agreement ratio (P_O) should be computed (Equation 4.14). In our case, it is the sum of P_{NN} and P_{RR}, since we want our algorithm to find road points as road points and non-road points as non-road points. Moreover, expected value (P_E) of the case in which the voters are totally independent should be calculated (Equation 4.15). After these calculations, the kappa coefficient (κ) can be calculated as in equation 4.16.

$$P_O = P_{RR} + P_{NN} \tag{4.14}$$

$$P_E = P_{\_R} \cdot P_{R\_} + P_{\_N} \cdot P_{N\_} \tag{4.15}$$

$$\kappa = \frac{P_O - P_E}{1 - P_E} \tag{4.16}$$

Kappa coefficient is always smaller than 1. The interpretation of kappa coefficient is given in Table 4-12.

Table 4-12: Kappa coefficient interpretation

$\kappa < 0.20$	Poor agreement
$0.20 \le \kappa < 0.40$	Fair agreement
$0.40 \le \kappa < 0.60$	Moderate agreement
$0.60 \le \kappa < 0.80$	Good agreement
$0.80 \le \kappa < 1$	Very good agreement
$\kappa = 1$	Perfect agreement

For all methods using 720 test images, Measure A, Measure B and Measure C results are obtained. Then the mean and standard deviation of all measures are calculated. The results for all measure and training durations are tabulated in Table 4-13. In Table 4-13, color codes are assigned to the tabulated to the mean and standard deviation values. Green colors represent relatively successful results, whereas red colors represent poorer results. On the other hand yellow colors represent moderately successful results. Moreover separate measure comparisons are given in Figure 4-25, Figure 4-26 and Figure 4-27.

It can be clearly seen that among 33 ANN schemes, three ANNs which are trained by input features consisting of saturation and value channels of HSV color space and using a 8-bin independent histogram outperform. These schemes are methods with the following codes: SAT+VAL_k11_i8, SAT+VAL_k15_i8 and SAT+VAL_k31_i8. These methods have an excellent road detection performance. They have found road regions correctly with ratios of 96.3%, 96.4% and 96.7%, respectively. The standard deviations for these ratios are very small (3.4% for all of three) when compared to that of other methods. Moreover, for those there methods the ratios and standard deviations of incorrectly detected road points are relatively small. Finally, when the kappa coefficients of these three methods, which are 0.885, 0.888 and 0.888 respectively, are examined; these methods outperform all of the other methods without a doubt.

Another interesting point among these three methods is that, it can be observed that their results are kernel size invariant. In other words, the

change in the size of the kernel from which the vector information is extracted, the performance of the method is not affected.

In CHAPTER 5, using these road detection results, reference robot heading is evaluated. While computing reference robot heading using the above mentioned three methods, desired results are not obtained. After a detailed examination is carried out and it is seen that computed road profiles are very similar to each other. It means that, in the methods fed (x,y) coordinates dominates the input feature vectors and suppress the other saturation and value information in the vectors. Therefore, these methods are eliminated.

After eliminating top three networks, among remaining 30 ANN schemes, three ANNs, which are trained by totally different input features, prevail. These schemes are methods with the following codes: SAT _k15_ALL, RGB+SAT_k15_i16 and RGB_k31_j4_i8. These methods have a pretty good road detection performance. They have found road regions correctly with ratios of 98.2%, 97.8% and 96.7%, respectively. The standard deviations for these ratios are relatively small (3.1%, 2.1% and 3.2% respectively) when compared to that of other methods. Moreover, for those there methods the ratios and standard deviations of incorrectly detected road points are relatively small. Finally, when the kappa coefficients of these three methods, which are 0.876, 0.867 and 0.871 respectively, are examined; these methods outperform all of the other methods.

Table 4-13: All performance Measures in (%) (κ=kappa coefficient, CDP = correctly detected points, IDP = incorrectly detected points)

METHOD CODE	κ (mean)	κ (std)	CDP (mean)	CDP (std)	IDP (mean)	IDP (std)	Train time
RGB k11 ALL	85.7	6.6	92.5	6.8	5.6	5.8	6.5
RGB k15 ALL	79.0	15.0	82.7	14.9	2.1	2.9	8.6
SAT k11 ALL	83.2	9.8	89.6	9.0	5.2	4.8	7.7
SAT k15 ALL	87.6	5.1	98.2	3.1	11.0	7.5	8.0
SAT+VAL k11 ALL	86.3	5.4	91.6	4.3	4.1	4.1	8.5
SAT+VAL k15 ALL	87.1	6.4	93.4	5.2	4.8	4.1	7.2
RGB+SAT k11 ALL	84.6	10.1	89.7	10.0	3.9	4.9	5.2
RGB+SAT k15 ALL	84.3	9.4	88.5	9.5	2.9	3.8	5.5
RGB k11 i8	78.9	12.6	90.0	12.4	8.4	8.4	6.6
RGB k15 i8	71.7	19.0	79.7	19.5	6.1	7.0	4.5
RGB k31 i8	73.8	18.6	78.6	20.1	7.3	7.5	8.7
SAT k11 i8	86.0	6.2	94.8	5.5	7.7	6.6	6.1
SAT k15 i8	86.0	6.7	93.0	5.8	5.9	5.9	6.5
SAT k31 i8	84.3	7.7	93.6	6.6	8.0	7.5	6.8
SAT k11 i16	84.4	9.9	90.5	9.2	4.7	5.2	6.2
SAT k15 i16	83.9	13.5	91.0	12.0	6.0	6.3	5.9
SAT k31 i16	80.0	11.3	92.9	12.0	11.5	8.7	7.1
SAT+VAL k11 i8	88.5	4.9	96.3	3.4	6.9	5.9	5.5
SAT+VAL k15 i8	88.8	5.0	96.4	3.4	6.7	5.9	4.9
SAT+VAL k31 i8	88.8	5.1	96.7	3.4	7.0	6.0	4.8
RGB+SAT k11 i8	78.5	9.6	87.0	10.2	6.7	6.7	6.2
RGB+SAT k15 i8	80.5	9.2	85.4	18.3	7.6	7.6	6.6
RGB+SAT k31 i8	75.0	19.6	84.3	19.7	7.8	9.2	7.2
RGB+SAT k11 i16	82.4	6.5	94.0	5.7	7.5	6.1	6.3
RGB+SAT k15 i16	86.7	5.9	97.8	2.1	8.2	7.1	6.7
RGB+SAT k31 i16	78.2	9.0	84.2	9.2	4.2	5.1	7.3
RGB k11 j4 i8	74.7	17.7	84.6	19.0	7.5	8.2	6.2
RGB k15 j4 i8	80.3	10.5	92.7	5.0	10.2	10.3	7.4
RGB k31 j4 i8	87.1	4.9	96.7	3.2	7.4	6.2	5.3
RGB k11 j4 i16	79.4	18.6	88.6	19.5	8.0	7.0	7.7
RGB k15 j4 i16	71.5	29.2	81.1	31.2	8.4	7.3	7.4
RGB k31 j4 i16	78.4	17.8	87.8	17.9	7.3	6.5	8.2
SAT+VAL k15 j4 i8	69.6	21.4	84.1	21.2	10.5	10.5	7.3

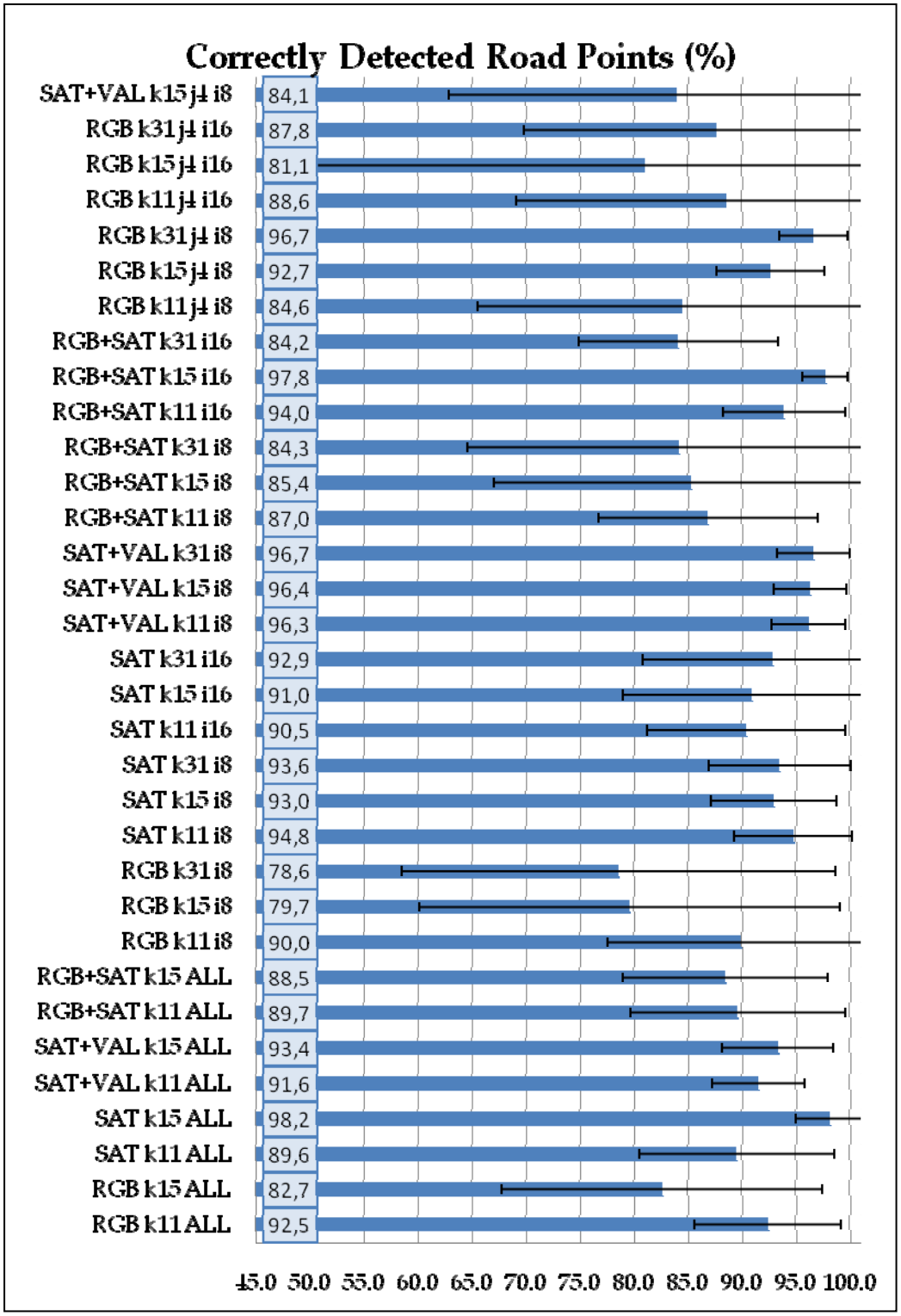

Figure 4-25: Road detection performance of ANN's fed by different input feature vectors according to Measure – A

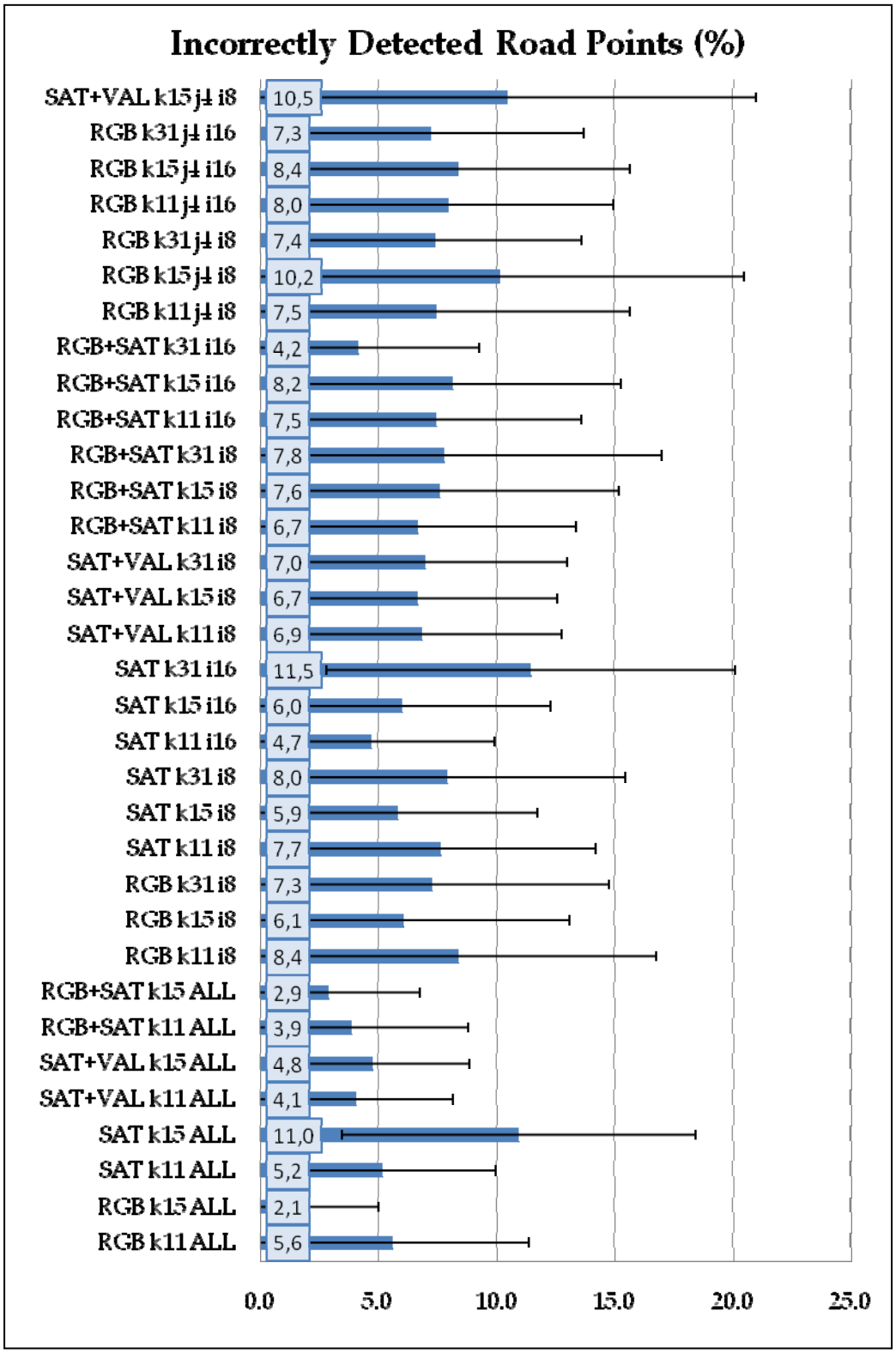

Figure 4-26: Road detection performance of ANN's fed by different input feature vectors according to Measure – B

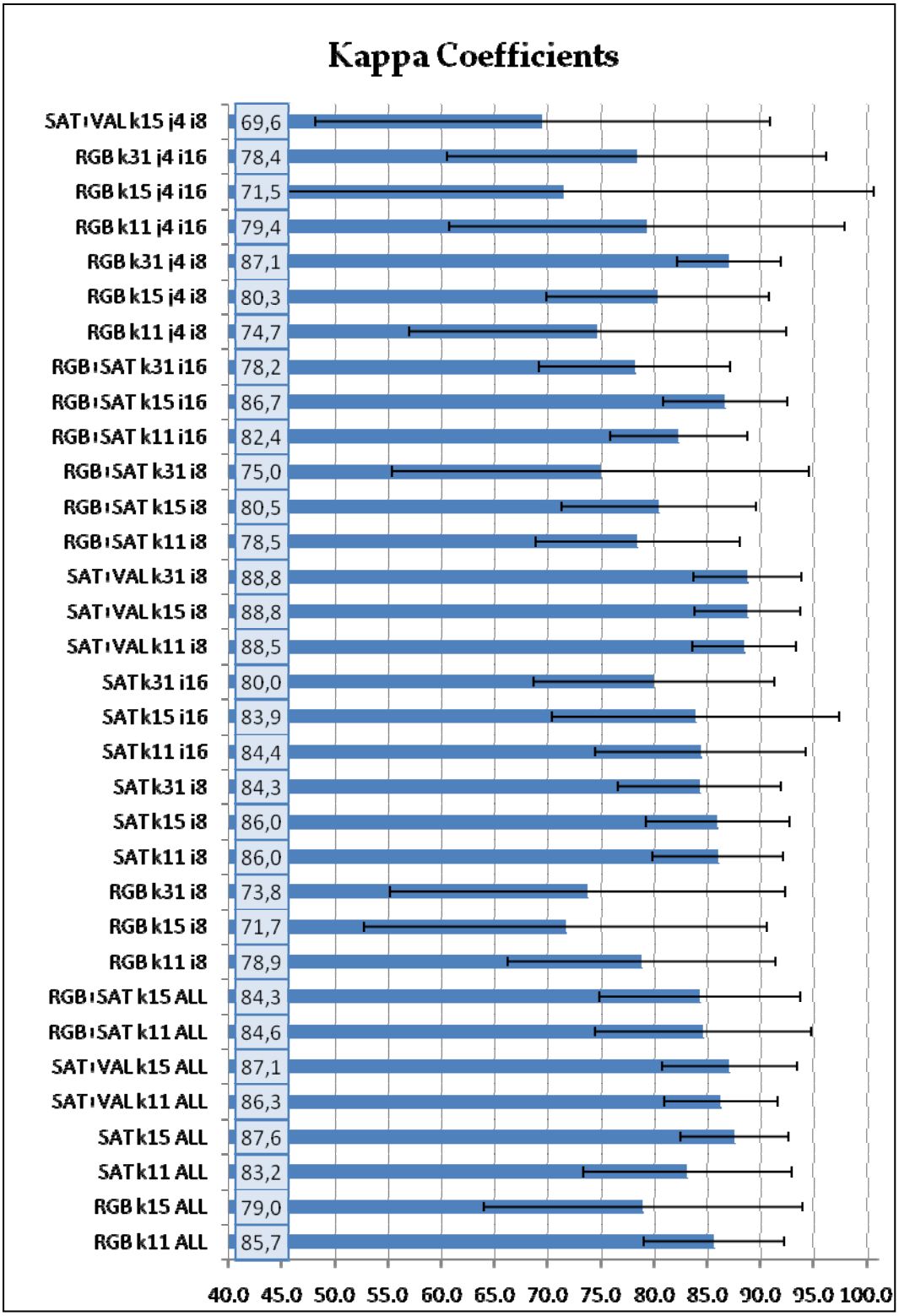

Figure 4-27: Kappa coefficients for performance of ANNs fed by different input feature vectors according (Measure – C)

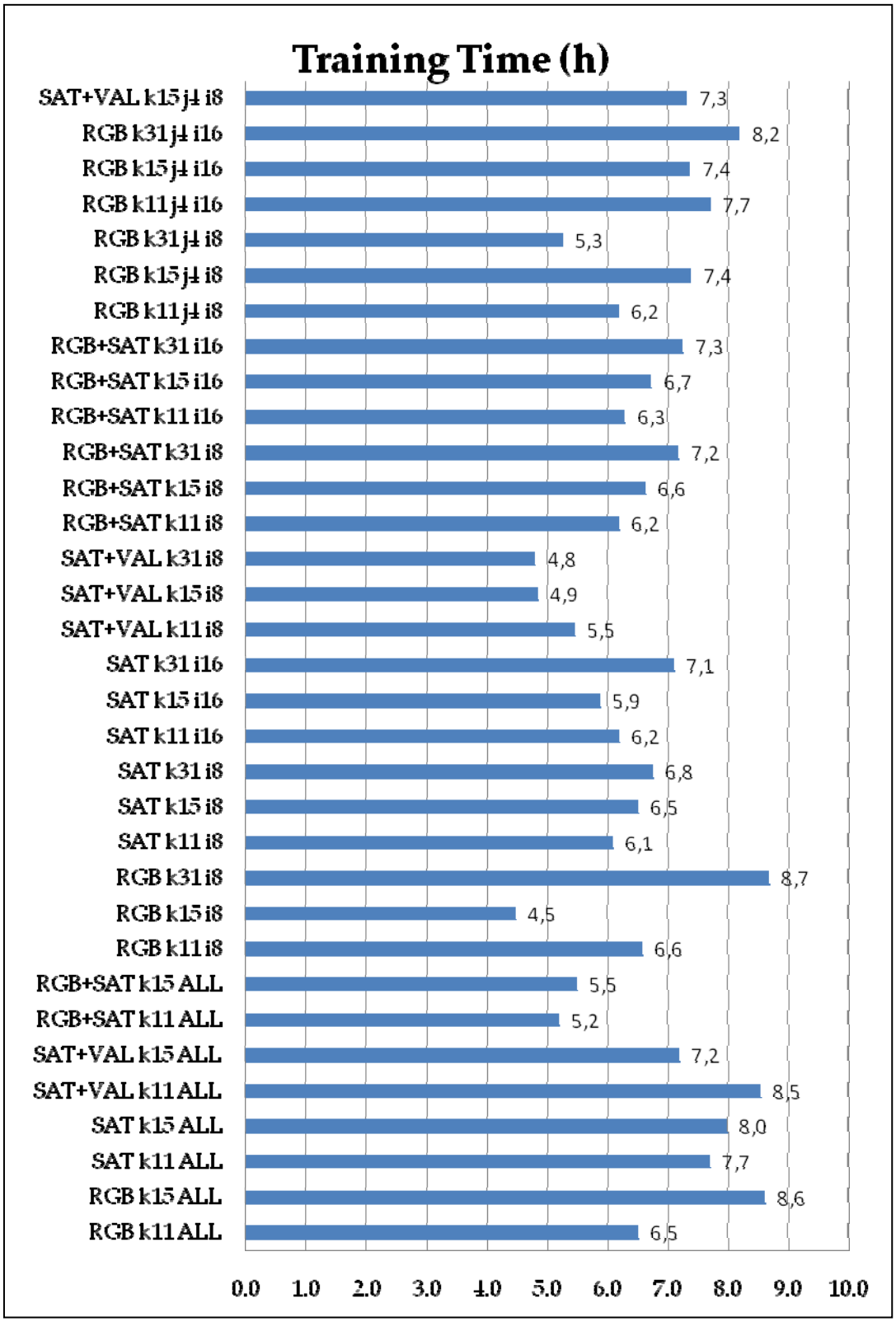

Figure 4-28: Training times of ANN's

Among these three methods, the most successful method is RGB+SAT_k15_i16 method. Since even if the kappa coefficient results are very near to each other, RGB+SAT_k15_i16 outperforms the other two methods when the computed reference robot heading vectors are compared. The other two methods found out to be dominated by (x,y) coordinate values included in the input feature vectors, when the resultant road profiles are examined. However, in RGB+SAT_k15_i16 the information obtained through color channels seemed to be utilized since resultant road profiles are different for different road inputs.

Finally, training durations for all of ANN training schemes are depicted in Figure 4-28. All ANN trainings are held at a computer having an Intel® Core™ i7 CPU 920 2.79GHz processor and a 8 GB RAM. When the training times are compared, it can be seen that training time of RGB+SAT_k15_i16 is 6.7 hours which is below the average. Moreover 6.7 hours is acceptable for an input database of 720 images.

4.4.7 Conclusion

In this section, a road recognition algorithm utilizing ANNs and color-based feature vectors is discussed. All types of feature vectors and their generation procedure are given. Sample outputs and detailed results of trained ANNs are presented.

The results assure that the ANN training scheme utilizing RGB+SAT_k15_i16 method as input feature vector outperforms the 32 other

schemes. Therefore, this scheme will be used in order to compute the reference heading of the robot, which is presented in CHAPTER 5.

It is very interesting to see that the method which turned out to be the best among the other methods only uses RGB channels of RGB color space with saturation channel of HSV color space.

CHAPTER 5

COMPUTATION OF REFERENCE ROBOT HEADING

5.1 Introduction

After the best configurations for the proposed methods are selected, a reference robot heading, which gives information to the vehicle about the direction to be headed to, will be computed using the binary image obtained at the output of the trained ANN with the chosen configuration. This computed reference robot heading is then compared with the computed vanishing point in the same image. According to this comparison, it is decided whether the computed reference robot heading is valid or not.

In this chapter, the outputs of trained ANN for 144 test images, which are the same test images used in evaluation of vanishing point algorithm, are taken and a reference robot heading is computed for each of these images. The validity of the reference robot heading will be tested with the corresponding vanishing point. Then, the performance of the routine extracting the reference robot heading will be evaluated by comparing the results with reference vectors generated manually. Finally, the results will be discussed, and comments on the possible improvements and future works that may be carried on this subject will be given.

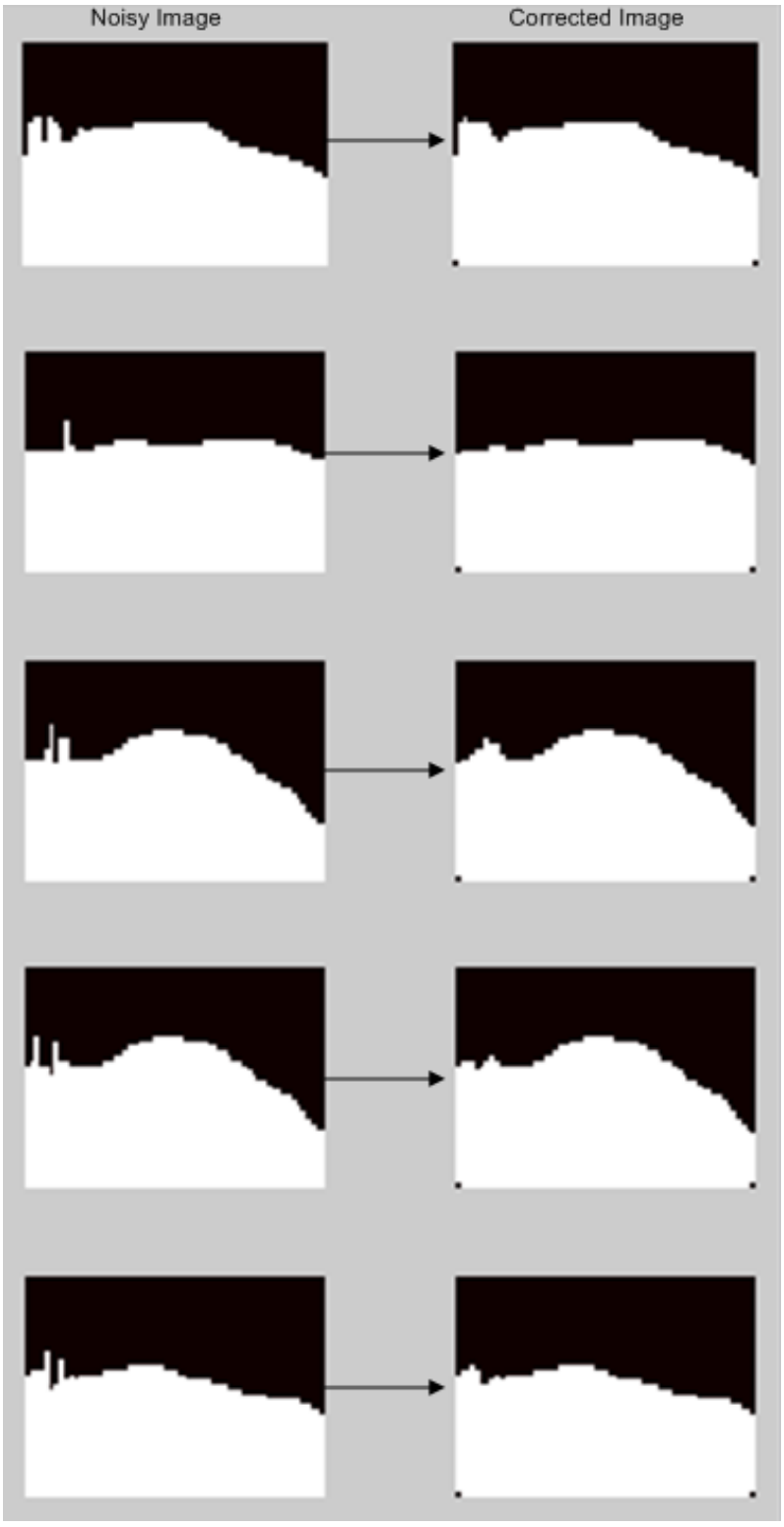

Figure 5-1: Examples of noisy and corrected images

5.2 Pre-processing of ANN Results

When the outputs of the trained ANN for the mentioned test image set, it is seen that there are noises at the right portion of the binary result images. Examples of this kind of noisy images are depicted in noisy image column of Figure 5-1. These noisy parts in the result images presumably will cause reference robot heading computation results to deteriorate. For this sake, the results should be edited using binary morphological operations.

There are two types of defects observed in the result images. These are the noisy parts appear in the right of the image and the vacancy pixels within the road area.

Noisy parts can be smoothened using simple median filter. Working principle of 3 x 3 size median filter is depicted in Figure 5-2. When median filter window is placed on a portion of image, the values of pixels falling within that window are taken and then sorted. Value of the middle element of this sorted array is placed into the pixel where the median filter is placed as the result of the filtering operation.

Moreover, vacancy points in the result images can simply eliminated using morphological closing operation. Closing operation can be defined morphological dilation operation followed by morphological erosion operation. Closing operation smoothes the contours of binary images, eliminates small holes and finally fills gaps in the contour. Dilation and erosion operations applied on the same image with the same structuring element (disc) are depicted in Figure 5-3.

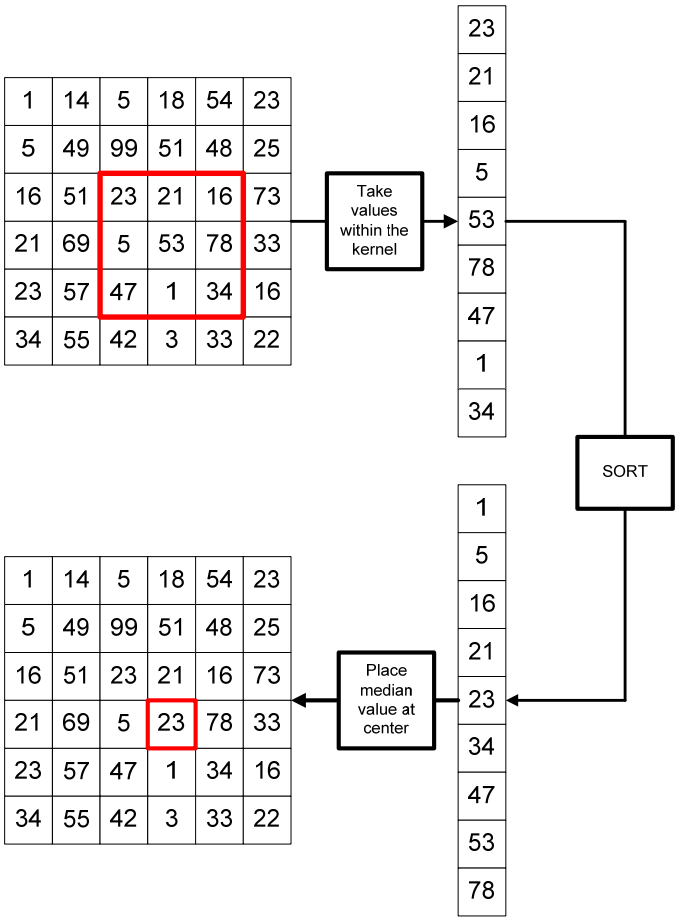

Figure 5-2: Operation algorithm of median filter

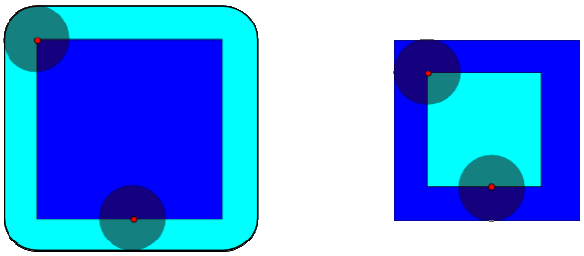

Figure 5-3: (a)Dilation[4] - (b) Erosion[5]

[4] http://en.wikipedia.org/wiki/File:Dilation.png
[5] http://en.wikipedia.org/wiki/File:Erosion.png

In order to improve result images, first of all closing operation is applied with structure element '+' to all result images. Then several median filters whose sizes changes from 1x1 to 5x5 are applied to the image and it is seen that reference heading results computed from results image corrected with 3 x 3 median filter is the most successful one. Therefore, 3 x3 median filter is used for correcting the result images.

Examples for corrected result images are depicted in corrected image column of Figure 5-1. All of the test images are corrected using the mentioned operations and fed to the algorithm computing the reference robot heading. The details of the algorithm are explained in section 5.3.

5.3 Determination of Reference Robot Heading

After the improvement of outputs of trained ANN, a reference robot heading should be computed in order to lead the mobile robot to the safe direction. The pseudo code for the proposed algorithm is given in Table 5-1

For the reference robot heading computation, the binary result image is divided into horizontal bands with a height of 10 pixels. Starting the bottommost band, the center of the road area is computed. This routine goes on for every stripe until one of the two stop criteria is met. These criteria are:

- No road region within the horizontal band
- Multiple road regions within the horizontal band

After the centers of road regions in each horizontal band are obtained, using a linear least square fitting [87], a linear polynomial is fitted to these coordinates in order to construct the reference heading vector.

Following that, the computed reference heading is compared with the computed vanishing point with in the same test image. If vector pointing to the vanishing point is close enough to the computed heading, the computed heading is said to be reliable. Otherwise, the computed heading is said to be unreliable and eliminated. Four examples of computed reference headings can be seen in Figure 5-4 and Figure 5-5. The binary image at the left top corner is the corrected binary result. Piecewise lines are the connections between centers of the horizontal strips. Line with asterisk depicts the direction of the heading.

Table 5-1: Pseudo code for reference robot heading computation

```
FOR strip number = last strip TO first strip
      LABEL the connected regions in the strip
      IF # of regions >1 THEN
            BREAK loop
      ELSE
            FIND the midpoint of the road region in the strip
            STORE the abscissa to the abscissaeMatrix
            STORE strip number to the ordinatesMatrix
      ENDIF
ENDFOR
FIT a linear polynomial using abscissaeMatrix and ordinatesMatrix
HEADING VECTOR = fitted linear polynomial
AngleDiff = ABS(heading_angle – vanish_angle)
IF AngleDiff < Threshold_Degree
      USE Heading Vector
ELSE
      ELIMINATE Heading Vector
ENDIF
```

Figure 5-4: Computed reference headings – 1

Figure 5-5: Computed reference headings – 2

5.4 Experiments & Results

In this section, the evaluation of the performance of the reference robot heading computation algorithm is performed. In order to evaluate the performance of the algorithm, truth reference robot heading directions were generated manually for 144 test images.

The performance of the method is evaluated by taking the dot product of truth reference headings with computed reference headings. Then histogram of dot product results with 0.01 resolutions is generated in order to see the correctly detected heading distribution. This result is depicted in Figure 5-6.

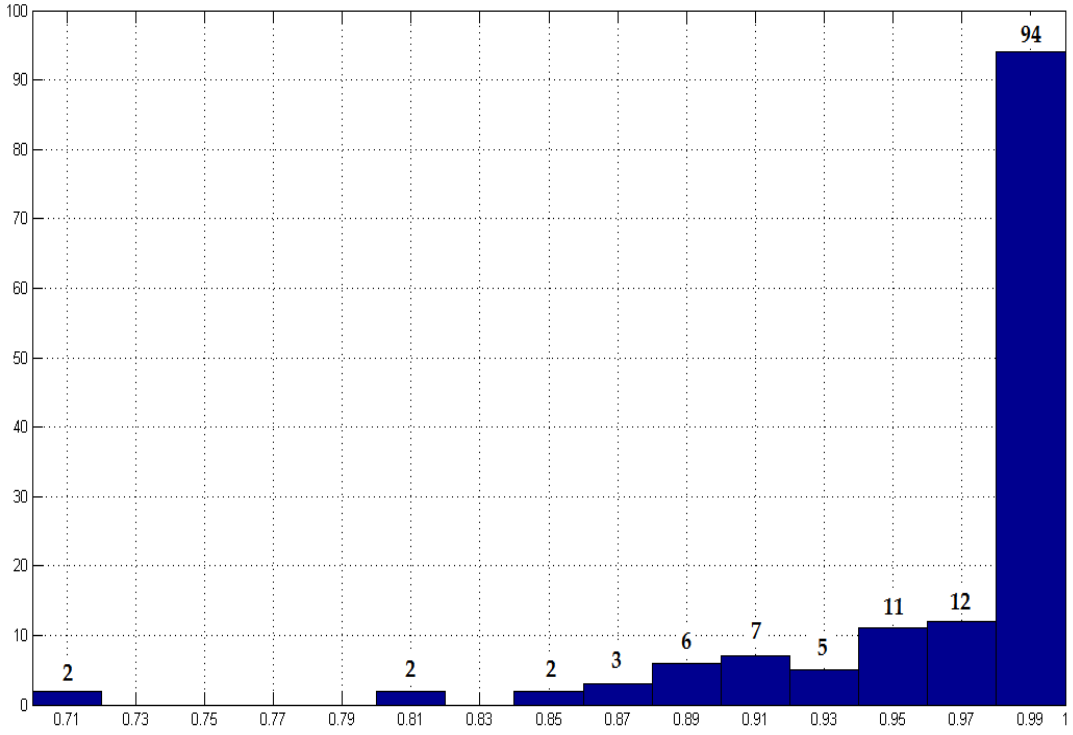

Figure 5-6: Computed & truth headings dot product histogram

As can be seen in the histogram, in 94 test images (65%) the angle between computed reference heading and the truth heading is smaller than 8 degrees since their dot product is greater than 0.99. Moreover in 106 test images (74%) this angle is smaller than 14 degrees. By examining this histogram the performance of the algorithm can be presented as 65%. However, algorithm presents that the computed reference headings are valid only if they are in accordance with the corresponding vanishing points. Therefore, dot products of vanishing point vectors and computed reference headings should be taken and the computed reference heading which are not highly correlated with vanishing point locations should be eliminated before the performance evaluation. The related histogram showing the validity of the computed reference headings are given in Figure 5-7.

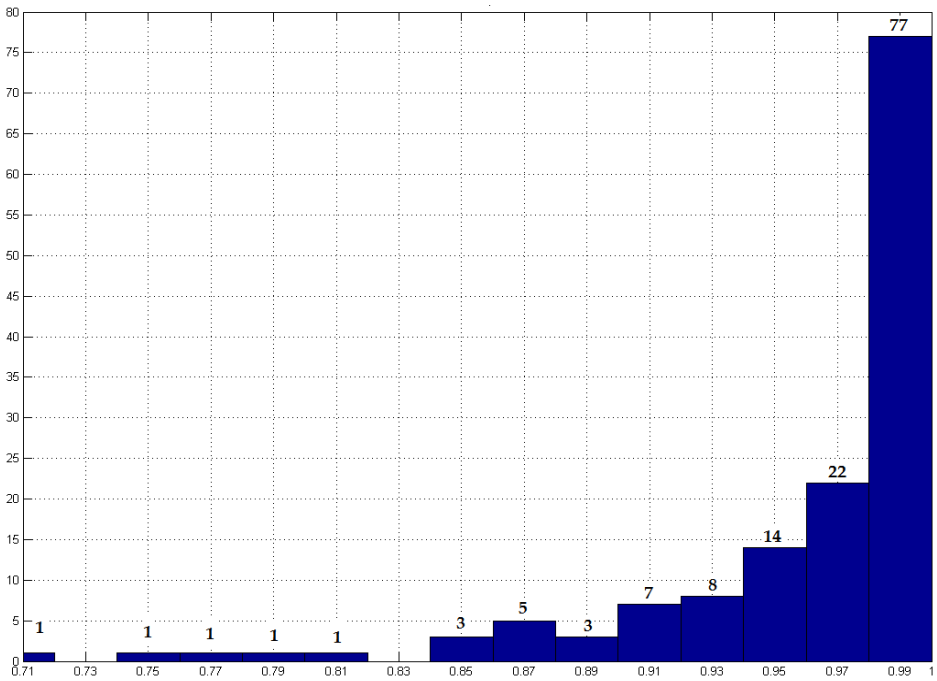

Figure 5-7: Computed heading & vanishing point dot product histogram

As can be seen in Figure 5-7, the number of computed reference headings which are in the range of 8 degrees of vanishing point vectors is 77 (54% of all computed headings). So the other headings should be eliminated and should not be used as a direction reference to the robot. For obtaining robot reference heading for these eliminated images, the frames following those images should be used.

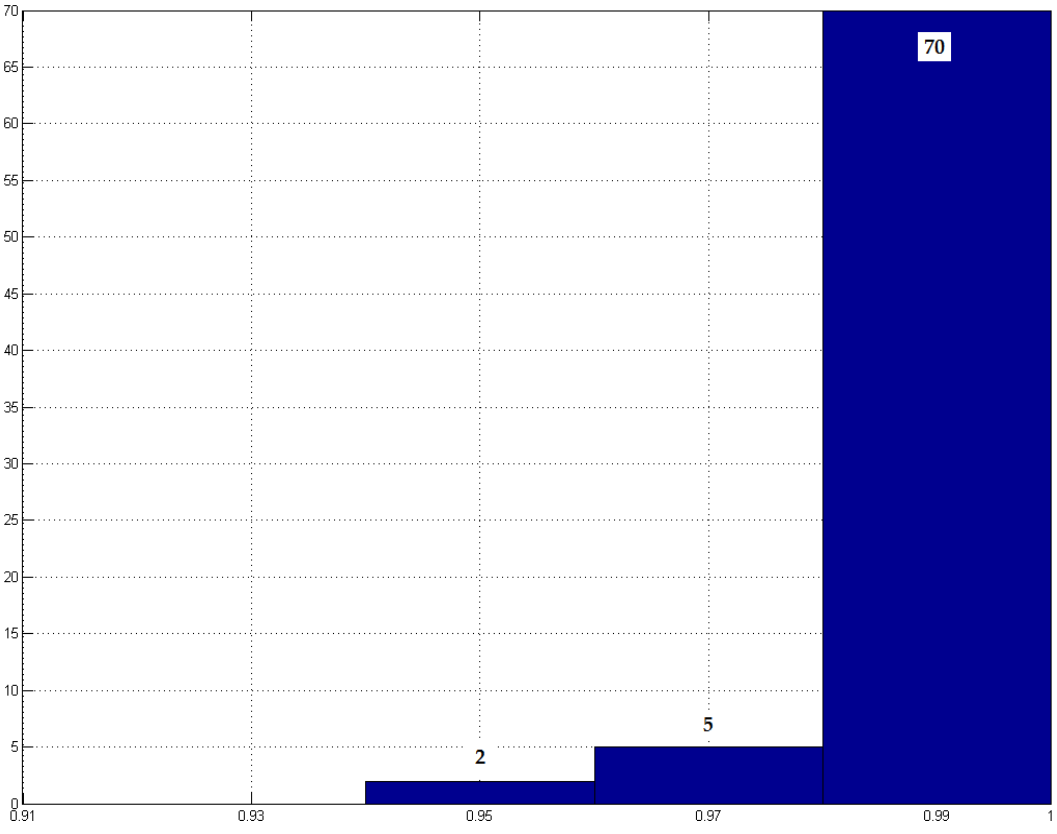

Figure 5-8: Computed & truth headings dot product histogram (with bad results are eliminated with vanishing point test)

When the computed reference headings which are not highly correlated are eliminated, 77 headings remain. Then dot product of these computed reference headings with the truth heading counterparts are calculated and a dot product histogram with a 0.01 resolutions is generated. This histogram is depicted in Figure 5-8.

When the histogram is examined, it is observed that for 70 computed reference headings (91%), the angle difference between the computed and truth headings is smaller than 8 degrees. While 5 computed reference headings (6.5%), this angle is between 8 degrees and 14 degrees. Therefore, the algorithm is said to compute nearly 90% of the reference headings correctly.

5.5 Conclusion

In this chapter, a routine is proposed in order to compute a reference robot heading direction by using road detection algorithm and vanishing point algorithm with the best configurations decided in CHAPTER 4. Routine utilizes the outputs of the trained ANN. First of all, the noisy outputs are improved using a median and a couple of morphological operation. Then, center points of road strips are computed step by step. Following that a linear polynomial, which turns out to be the heading vector, is fitted to the center points. This vector is compared with the vanishing point vector. If the heading vector is in the vicinity of the vanishing point, it is said to be a valid vector and it can be used by the mobile robot. Otherwise, the heading vector is said to be invalid and mobile robot should search for a valid

heading vector in the next frames. Overall routine proposed in this study is depicted in Figure 5-9.

Proposed routine has a robust structure since it can eliminate the heading vector results which have a greater probability to be wrong. Moreover, the valid heading vectors have a 90% of accuracy. It should be noted that, these results are obtained only using the information extracted from a single webcam, and other means of sensors are not utilized for gathering further information.

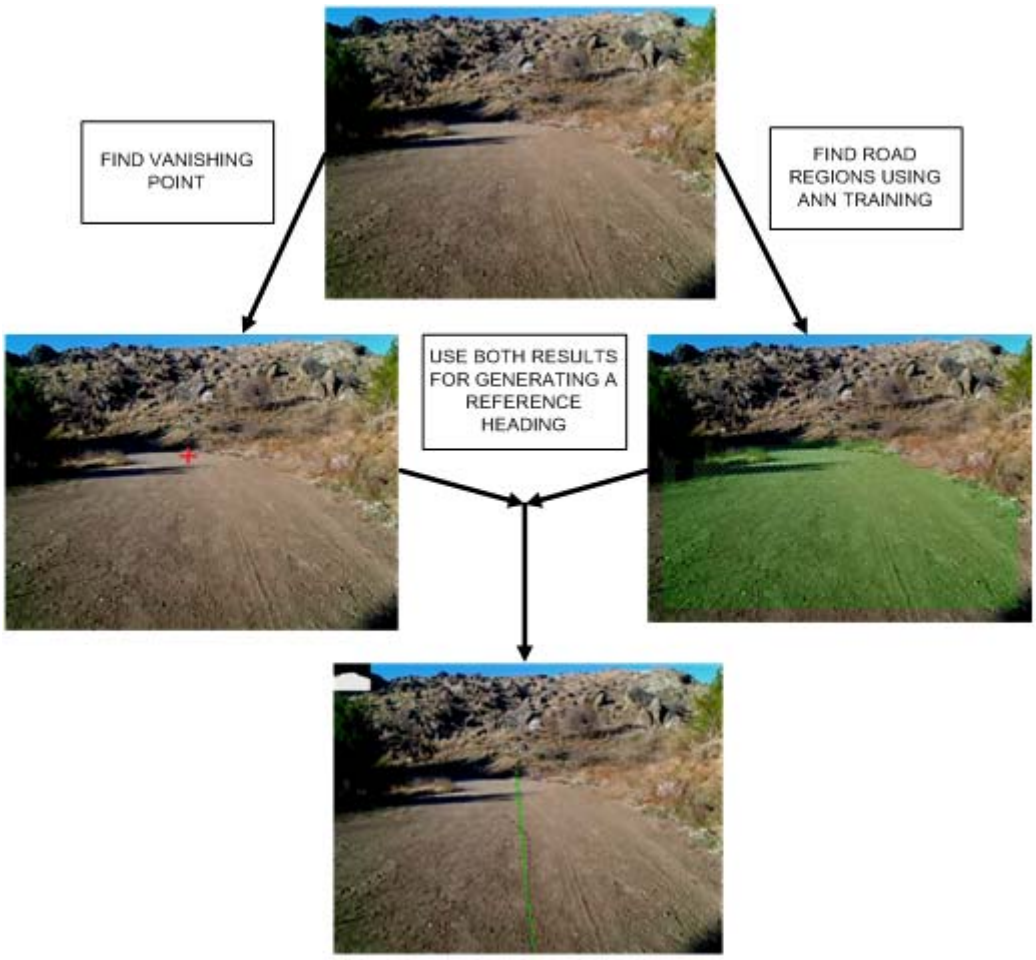

Figure 5-9: Overall view of the proposed method

CHAPTER 6

CONCLUSION & FUTURE WORK

6.1 Conclusion

In this study an assortment of road settings including several unstructured roads types is selected as target. For detection, images collected from simple webcam mounted on the mobile robot are used as the sole source of information to the robot. By processing these images, for any type of unstructured road, the robot should obtain the boundaries of the road and a reference robot heading representing the direction to be followed by the robot. The robot would not receive any information about its environment by any means of equipped sensors other than a simple camera.

First of all, very detailed reviews of the road detection techniques that are available in the literature are investigated. The dynamics of various road detection algorithms developed for both structured and unstructured roads are examined. Advantages and disadvantages of all examined method are listed and during the algorithm development phase these lists are utilized.

Unstructured road videos are recorded to be used in the development of the algorithm and test purposes. Videos of unstructured roads around the lake in the Middle East Technical University Campus are captured with VGA resolution (640x480) via a webcam attached to the front of the ATV (All-Terrain Vehicle) while vehicle speed is within the range of 8-25 km/h.

At the end of 37-minute late morning video and a total of 49,723 video frames were recorded in the end. For training purposes, 24 different video sections containing 300 frames are cut with different illumination and road texture characteristics from 3 videos resulting in 24 small sized videos. Out of each 24 video, 30 frames were randomly selected, producing 720 frames for training purposes.

As a classification method for the proposed road recognition algorithm, ANN method is selected. Most common type of ANN is selected with a specified configuration. Back propagation algorithm is used for the selected feed-forward ANN scheme during the training of network. The activation function is selected as 'unipolar sigmoid'. After the classification scheme is selected, different input feature vector schemes are constructed and fed into different ANNs. Input feature vectors utilize information of RGB color space or HSV color space or both color spaces. Four different histogram schemes, namely independent, joint, combined and 'all', are used in order to generate different input feature vectors. In every feature vector, other than color information coordinates of feature vector within the road image also included. 33 different input vectors are used to train 33 different ANN. After all the ANNs are trained their outputs are evaluated.

It is seen that input feature vectors utilizing one or two channels of HSV color space does not provide distinctive information for the classification of the road regions, and coordinate information of the kernel included in the feature vector become dominant in the ANN training. Similarly, when only RGB color space information with independent histogram scheme is used, again coordinates become dominant in ANN training. In order to use RGB

color information effectively, joint histogram scheme should be used in the construction of input feature vectors. Moreover, using independent histogram scheme utilizing color channel information of both color space generates distinctive input feature vectors for road classification. After the evaluation of all 33 methods, one of the method utilizing all channels of RGB color space and Saturation channel of HSV color space is chosen as the best performer.

Another part of the study is spared for the vanishing point computation of the road. Vanishing point computation is performed for several angular resolutions and finally 1° resolution is selected as the best performer.

Both of the chosen methods are utilized for the reference robot heading computation. The computation is performed using the binary road/non-road data obtained from the output of trained ANN and the resultant heading is compared to the corresponding vanishing point result for questioning its validity.

After the reference robot heading computation results are evaluated, it is observed that the algorithm eliminates unreliable heading results robustly and the valid heading results nearly have a performance rating of 90%. This means that, algorithm, which uses a camera as the sole sensor collecting the information about the road, can be considered successful.

6.2 Future Work

In this study, several schemes for ANN training phase are considered. However, since the number of free parameters available within each scheme is enormous, all of the considered schemes cannot be applied. Only one of the schemes is selected and it is investigated. In the completed study, input feature vectors are extracted for road and non-road regions of the image and single ANN is trained using these input feature vectors.

Other than this feature vector generation and training scheme, an alternative training scheme was considered but cannot be put in practice. In this alternative training scheme contains multiple ANNs. In this scheme, non-road features are categorized into several groups. Examples of such groups can be tree, bushes, sand and sky. Input feature vectors (exclusive feature vectors) are extracted for each of these non-road groups and different ANNs are trained for each group. In the end, using the trained ANNs, sky, tree, bush and sand areas are revealed and the remaining areas are found as the road region. In further study, it is planned that this alternative ANN training scheme will be put into practice.

Moreover, in this study only a single VGA camera is used as the source of information in order to explore the boundaries of road recognition performance using only image data. However, in future studies additional sensors are planned to be used since the real objective of the road recognition is to allow the mobile robot to follow the path and explore the unknown terrains. Therefore, additional sensors are essential. These additional sensors may be omni-directional cameras, or multiple cameras

mounted on different locations of the mobile robot. Moreover, laser range sensors may be integrated into the mobile system to improve the obstacle avoidance performance. By integrating several other sensors on the mobile system, improved mobility will be tried to be accomplished.

Finally, the proposed algorithms are only simulated on a computer environment in this study. In the future, deploying algorithms to FPGAs or GPUs are planned to improve the performance of these algorithms running in real-time on a mobile robot.

REFERENCES

[1] Crisman, J. D., and Thorpe, C. E., "Scarf: A Color Vision System That Tracks Roads and Intersections," *Robotics and Automation, IEEE Transactions on*, vol. 9, no. 1, pp. 49-58, 1993.

[2] Behringer, R., "Road Recognition from Multifocal Vision," *Proceedings of Intelligent Vehicles '94 Symposium, Proceedings of the*, pp. 302-307, 1994.

[3] Sahli, H., De Muynck, P., and Cornelis, J., "A Kalman Filter-Based Update Scheme for Road Following," *Proceedings of Proceedings of IAPR Workshop on Machine Vision Applications*, pp. 5-9, Tokyo, Japan, 1996.

[4] Paetzold, F., and Franke, U., "Road Recognition in Urban Environment," *Image and Vision Computing*, vol. 18, no. 5, pp. 377-387, 2000.

[5] You, F., Wang, R.-B., and Zhang, R.-H., "Research on Road Recognition Algorithm Based on Structure Environment for Its," *Proceedings of Computing, Communication, Control, and Management, 2008. CCCM '08. ISECS International Colloquium on*, pp. 84-87, 2008.

[6] Chen, K.-H., and Tsai, W.-H., "Vision-Based Autonomous Land Vehicle Guidance in Outdoor Road Environments Using Combined Line and Road Following Techniques," *Journal of Robotic Systems*, vol. 14, no. 10, pp. 711-728, 1997.

[7] Jung, C. R., and Kelber, C. R., "Lane Following and Lane Departure Using a Linear-Parabolic Model," *Image and Vision Computing*, vol. 23, no. 13, pp. 1192-1202, 2005.

[8] Samadzadegan, F., Sarafraz, A., and Tabibi, M., "Automatic Lane Detection in Image Sequences for Vision Based Navigation Purposes," 2006.

[9] Liu, J., Li, Z., and Chen, T., "Study on Road Recognition Algorithm," *Proceedings of Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pp. 2539-2541, 2007.

[10] Aly, M., "Real Time Detection of Lane Markers in Urban Streets," *Proceedings of Intelligent Vehicles Symposium, 2008 IEEE*, pp. 7-12, 2008.

[11] Assidiq, A. A. M., Khalifa, O. O., Islam, R., and Khan, S., "Real Time Lane Detection for Autonomous Vehicles," *Proceedings of Computer and Communication Engineering, 2008. ICCCE 2008. International Conference on*, pp. 82-88, 2008.

[12] Aufrère, R., Chapuis, R., and Chausse, F., "A Model-Driven Approach for Real-Time Road Recognition," *Machine Vision and Applications*, vol. 13, no. 2, pp. 95-107, 2001.

[13] Aufrere, R., Marmoiton, F., Chapuis, R., Collange, F., and Derutin, J. P., "Road Detection and Vehicle Tracking by Vision for Adaptive Cruise Control," *The International Journal of Robotics Research*, vol. 20, no. 4, pp. 267-286, 2001.

[14] Chapuis, R., Aufrere, R., and Chausse, F., "Accurate Road Following and Reconstruction by Computer Vision," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 3, no. 4, pp. 261-270, 2002.

[15] Yinghua, H., Hong, W., and Bo, Z., "Color-Based Road Detection in Urban Traffic Scenes," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 5, no. 4, pp. 309-318, 2004.

[16] Lin, L., and Zhou, W., "A Robust and Adaptive Road Following Algorithm for Video Image Sequence," 2007.

[17] Liu, H.-J., Guo, Z.-B., Lu, J.-F., and Yang, J.-Y., "A Fast Method for Vanishing Point Estimation and Tracking and Its Application in Road Images," *Proceedings of ITS Telecommunications Proceedings, 2006 6th International Conference on*, pp. 106-109, 2006.

[18] Lipski, C., Scholz, B., Berger, K., Linz, C., Stich, T., and Magnor, M., "A Fast and Robust Approach to Lane Marking Detection and Lane Tracking," *Proceedings of Image Analysis and Interpretation, 2008. SSIAI 2008. IEEE Southwest Symposium on*, pp. 57-60, 2008.

[19] Crisman, J. D., and Thorpe, C. E., "Unscarf-a Color Vision System for the Detection of Unstructured Roads," *Proceedings of Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 2496-2501 vol.3, 1991.

[20] Hong, C., Nanning, Z., and Tie, L., "An Approach of Road Recognition Based Mean Shift and Feature Clustering," *Proceedings of Signal Processing, 2002 6th International Conference on*, pp. 1059-1062 vol.2, 2002.

[21] Apostoloff, N., and Zelinsky, A., "Robust Vision Based Lane Tracking Using Multiple Cues and Particle Filtering," *Proceedings of Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, pp. 558-563, 2003.

[22] Kreucher, C., and Lakshmanan, S., "A Frequency Domain Approach to Lane Detection in Roadway Images," *Proceedings of Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, pp. 31-35 vol.2, 1999.

[23] Okutomi, M., and Noguchi, S., "Extraction of Road Region Using Stereo Images," *Proceedings of Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, pp. 853-856 vol.1, 1998.

[24] Alvarez, J. M., Lopez, A., and Baldrich, R., "Illuminant-Invariant Model-Based Road Segmentation," *Proceedings of Intelligent Vehicles Symposium, 2008 IEEE*, pp. 1175-1180, 2008.

[25] Younguk, Y., and Se-Young, O., "Three-Feature Based Automatic Lane Detection Algorithm (Tfalda) for Autonomous Driving," *Proceedings of Intelligent Transportation Systems, 1999. Proceedings. 1999 IEEE/IEEJ/JSAI International Conference on*, pp. 929-932, 1999.

[26] Ming-Yang, C., "Knowledge-Based Region Classification for Rural Road Area in the Color Scene Image," *Proceedings of Networking, Sensing and Control, 2004 IEEE International Conference on*, pp. 891-896 Vol.2, 2004.

[27] Pomerleau, D. A., "Efficient Training of Artificial Neural Networks for Autonomous Navigation," *Neural Computation*, vol. 3, no. 1, pp. 88-97, 1991.

[28] Jochem, T. M., Pomerleau, D. A., and Thorpe, C. E., "Maniac: A Next Generation Neurally Based Autonomous Road Follower," *Proceedings of International Conference on Intelligent Autonomous Systems*, Pittsburgh, Pennsylvania, USA, 1993.

[29] Jochem, T. M., Pomerleau, D. A., and Thorpe, C. E., "Vision-Based Neural Network Road and Intersection Detection and Traversal," *Proceedings of Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, pp. 344-349 vol.3, 1995.

[30] Rasmussen, C., "Combining Laser Range, Color, and Texture Cues for Autonomous Road Following," *Proceedings of Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, pp. 4320-4325 vol.4, 2002.

[31] Conrad, P., and Foedisch, M., "Performance Evaluation of Color Based Road Detection Using Neural Nets and Support Vector Machines," *Proceedings of Applied Imagery Pattern Recognition Workshop, 2003. Proceedings. 32nd*, pp. 157-160, 2003.

[32] Se-Young, O., Jeong-Hoon, L., and Doo-Hyun, C., "A New Reinforcement Learning Vehicle Control Architecture for Vision-Based Road Following," *Vehicular Technology, IEEE Transactions on*, vol. 49, no. 3, pp. 997-1005, 2000.

[33] Wei, L., Xiaojia, J., and Yangxing, W., "Road Recognition for Vision Navigation of an Autonomous Vehicle by Fuzzy Reasoning," *Fuzzy Sets and Systems*, vol. 93, no. 3, pp. 275-280, 1998.

[34] Ma, B., Lakahmanan, S., and Hero, A., "Road and Lane Edge Detection with Multisensor Fusion Methods," *Proceedings of Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, pp. 686-690 vol.2, 1999.

[35] Yanqing, W., Deyun, C., and Chaoxia, S., "Vision-Based Road Detection by Adaptive Region Segmentation and Edge Constraint," *Proceedings of Intelligent Information Technology Application, 2008. IITA '08. Second International Symposium on*, pp. 342-346, 2008.

[36] Yanqing, W., C., D., S., C., and W., P., "Vision-Based Road Detection by Monte Carlo Method," *Information Technology Journal*, vol. 9, no. 3, pp. 481-487, 2010.

[37] Hyongsuk, K., Seungwan, H., Hongrak, S., Roska, T., and Werblin, F., "High Speed Road Boundary Detection on the Images for Autonomous Vehicle with the Multi-Layer Cnn," *Proceedings of Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on*, pp. V-769-V-772 vol.5, 2003.

[38] Haiping, L., Hyongsuk, K., Chun-Shin, L., and Chua, L. O., "Road Boundary Detection Based on the Dynamic Programming and the Randomized Hough Transform," *Proceedings of Information Technology Convergence, 2007. ISITC 2007. International Symposium on*, pp. 63-70, 2007.

[39] Haiping, L., Suhong, K., Hyongsuk, K., and Yeongmin, K., "Road Boundary Detection with Double Filtering for Intelligent Vehicle," *Proceedings of Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on*, pp. 686-690, 2007.

[40] Nourine, R., Boudihir, M. E., and Khelifi, S. F., "Application of Radon Transform to Lane Boundaries Tracking," *Image Analysis and Recognition*, pp. 563-571, 2004.

[41] Zezhong, X., "Laser Rangefinder Based Road Following," *Proceedings of Mechatronics and Automation, 2005 IEEE International Conference*, pp. 713-717 Vol. 2, 2005.

[42] Chewputtanagul, P., and Jackson, D. J., "A Road Recognition System Using Gps/Gis Integrated System," *Proceedings of TENCON 2004. 2004 IEEE Region 10 Conference*, pp. 225-228 Vol. 4, 2004.

[43] Jian, W., Zhong, J., and Yu-Ting, S., "Unstructured Road Detection Using Hybrid Features," *Proceedings of Machine Learning and Cybernetics, 2009 International Conference on*, pp. 482-486, 2009.

[44] Ekinci, M., Gibbs, F., and Thomas, B. T., "Knowledge-Based Navigation for Autonomous Road Vehicles," *Turkish J. of Elec. Eng. & Comp. Sci*, vol. 8, pp. 1-29, 2000.

[45] Jansen, P., Van Der Mark, W., Van Den Heuvel, J. C., and Groen, F. C. A., "Colour Based Off-Road Environment and Terrain Type Classification," *Proceedings of Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pp. 216-221, 2005.

[46] Nefian, A. V., and Bradski, G. R., "Detection of Drivable Corridors for Off-Road Autonomous Navigation," *Proceedings of Image Processing, 2006 IEEE International Conference on*, pp. 3025-3028, 2006.

[47] Dahlkamp, H., "Selfsupervised Monocular Road Detection in Desert Terrain," *Proceedings of In Proc. of Robotics: Science and Systems* 2006.

[48] Tan, C., Tsai, H., Chang, T., and Shneier, M., "Color Model-Based Real-Time Learning for Road Following," *Proceedings of Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, pp. 939-944, 2006.

[49] Zhang, A. M., and L., K., "A Panoramic Color Vision System for Following Ill-Structured Roads " Auckland, 2006.

[50] Zheng, L., Bin, D., and Hangen, H., "A Novel Fast Segmentation Method of Unstructured Roads," *Proceedings of Vehicular Electronics and Safety, 2006. ICVES 2006. IEEE International Conference on*, pp. 53-56, 2006.

[51] Minghao, H., Wenjie, Y., Mingwu, R., and Jingyu, Y., "A Vision Based Road Detection Algorithm," *Proceedings of Robotics, Automation and Mechatronics, 2004 IEEE Conference on*, pp. 846-850 vol.2, 2004.

[52] Rasmussen, C., "Grouping Dominant Orientations for Ill-Structured Road Following," *Proceedings of Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, pp. I-470-I-477 Vol.1, 2004.

[53] Rasmussen, C., "Texture-Based Vanishing Point Voting for Road Shape Estimation," *Proceedings of British Machine Vision Conference*, 2004.

[54] Jingang, H., Bin, K., Bichun, L., and Fei, Z., "A New Method of Unstructured Road Detection Based on Hsv Color Space and Road Features," *Proceedings of Information Acquisition, 2007. ICIA '07. International Conference on*, pp. 596-601, 2007.

[55] Qingji, G., Qijun, L., and Sun, M., "Rough Set Based Unstructured Road Detection through Feature Learning," *Proceedings of Automation and Logistics, 2007 IEEE International Conference on*, pp. 101-106, 2007.

[56] Alon, Y., Ferencz, A., and Shashua, A., "Off-Road Path Following Using Region Classification and Geometric Projection Constraints," *Proceedings of Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, pp. 689-696, 2006.

[57] Konolige, K., Agrawal, M., Blas, M. R., Bolles, R. C., Gerkey, B., Sol\, J., \#224, and Sundaresan, A., "Mapping, Navigation, and Learning for Off-Road Traversal," *J. Field Robot.*, vol. 26, no. 1, pp. 88-113, 2009.

[58] Foedisch, M., and Takeuchi, A., "Adaptive Real-Time Road Detection Using Neural Networks," *Proceedings of Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*, pp. 167-172, 2004.

[59] Sun, J., Mehta, T., Wooden, D., Powers, M., Rehg, J., Balch, T., and Egerstedt, M., "Learning from Examples in Unstructured, Outdoor Environments," *Journal of Field Robotics*, vol. 23, no. 11-12, pp. 1019-1036, 2006.

[60] Erkan, A. N., Hadsell, R., Sermanet, P., Jan, B., Muller, U., and Lecun, Y., "Adaptive Long Range Vision in Unstructured Terrain," *Proceedings of Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pp. 2421-2426, 2007.

[61] Hadsell, R., Sermanet, P., Ben, J., Erkan, A., Han, J., Muller, U., and Lecun, Y., "Online Learning for Offroad Robots: Using Spatial Label

Propagation to Learn Long-Range Traversability," *Proceedings of Robotics: Science and Systems III*, 2007.

[62] Hadsell, R., Sermanet, P., Ben, J., Erkan, A., Scoffier, M., Kavukcuoglu, K., Muller, U., and Lecun, Y., "Learning Long-Range Vision for Autonomous Off-Road Driving," *Journal of Field Robotics*, vol. 26, no. 2, pp. 120-144, 2009.

[63] Lieb, D., Lookingbill, A., and Thrun, S., "Adaptive Road Following Using Self-Supervised Learning and Reverse Optical Flow," 2005.

[64] Poppinga, J., Birk, A., and Pathak, K., "Hough Based Terrain Classification for Realtime Detection of Drivable Ground," *Journal of Field Robotics*, vol. 25, no. 1-2, pp. 67-88, 2008.

[65] Wei, W., and Gong, S., "Research on Unstructured Road Detection Algorithm Based on the Machine Vision," *Proceedings of Information Processing, 2009. APCIP 2009. Asia-Pacific Conference on*, pp. 112-115, 2009.

[66] Ishida, H., Takahashi, T., Ide, I., Mekada, Y., and Murase, H., "Generation of Training Data by Degradation Models for Traffic Sign Symbol Recognition," *IEICE - Trans. Inf. Syst.*, vol. E90-D, no. 8, pp. 1134-1141, 2007.

[67] Jiang, F., Gao, W., Yao, H., Zhao, D., and Chen, X., "Synthetic Data Generation Technique in Signer-Independent Sign Language Recognition," *Pattern Recognition Letters*, vol. 30, no. 5, pp. 513-524, 2009.

[68] Yang, R., Sarkar, S., Loeding, B., and Karshmer, A., "Efficient Generation of Large Amounts of Training Data for Sign Language Recognition: A Semi-Automatic Tool," 2006.

[69] Cocosco, C. A., Zijdenbos, A. P., and Evans, A. C., "Automatic Generation of Training Data for Brain Tissue Classification from Mri," Springer-Verlag, 2002.

[70] Schwarz, D., and Kasparek, T., "Brain Tissue Classification with Automated Generation of Training Data Improved by Deformable Registration," 2007.

[71] Tam, V., "Generation of Synthetic Training Data for an Hmm-Based Handwriting Recognition System," *Proceedings* pp. 618-618, 2003.

[72] Monekosso, D., and Remagnino, P., "Synthetic Training Data Generation for Activity Monitoring and Behavior Analysis," Springer-Verlag, Salzburg, Austria, 2009.

[73] Jawahar, C. V., "Homography Estimation from Planar Contours," IEEE Computer Society, 2006.

[74] Gomez, D. D., Butakoff, C., Ersboll, B. K., and Stoecker, W., "Independent Histogram Pursuit for Segmentation of Skin Lesions," *Biomedical Engineering, IEEE Transactions on*, vol. 55, no. 1, pp. 157-161, 2008.

[75] Pass, G., and Zabih, R., "Comparing Images Using Joint Histograms," *Multimedia Syst.*, vol. 7, no. 3, pp. 234-240, 1999.

[76] Mcculloch, W., and Pitts, W., "A Logical Calculus of the Ideas Immanent in Nervous Activity," *Bulletin of Mathematical Biology*, vol. 5, no. 4, pp. 115-133, 1943.

[77] Blass, W. E., and Crilly, P. B., "An Introduction to Neural Networks Based on the Feed Forward, Backpropagation Error Correction Network with Weight Space Limiting Based on a Priori Knowledge," *Proceedings of Instrumentation and Measurement Technology Conference, 1992. IMTC '92., 9th IEEE*, pp. 631-634, 1992.

[78] Okada, R., Taniguchi, Y., Furukawa, K., and Onoguchi, K., "Obstacle Detection Using Projective Invariant and Vanishing Lines," *Proceedings of Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 330-337 vol.1, 2003.

[79] He, B. W., and Li, Y. F., "A Novel Method for Camera Calibration Using Vanishing Points," *Proceedings of Mechatronics and Machine Vision in Practice, 2007. M2VIP 2007. 14th International Conference on*, pp. 44-47, 2007.

[80] Rasmussen, C., "Roadcompass: Following Rural Roads with Vision + Ladar Using Vanishing Point Tracking," *Auton. Robots*, vol. 25, no. 3, pp. 205-229, 2008.

[81] Schuster, R., Ansari, N., and Bani-Hashemi, A., "Steering a Robot with Vanishing Points," *Robotics and Automation, IEEE Transactions on*, vol. 9, no. 4, pp. 491-498, 1993.

[82] Hui, K., Audibert, J. Y., and Ponce, J., "Vanishing Point Detection for Road Detection," *Proceedings of Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 96-103, 2009.

[83] Tai Sing, L., "Image Representation Using 2d Gabor Wavelets," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, no. 10, pp. 959-971, 1996.

[84] Mittal, N., Mital, D. P., and Kap Luk, C., "Features for Texture Segmentation Using Gabor Filters," *Proceedings of Image Processing And Its Applications, 1999. Seventh International Conference on (Conf. Publ. No. 465)*, pp. 353-357 vol.1, 1999.

[85] Danko, A., Alex, L., and Steven, D. J., "Vanishing Point: A Visual Road-Detection Program for a Darpa Grand Challenge Vehicle," 2005.

[86] *http://www.heatonresearch.com/articles/5/page2.html*, Last accessed on 27.05.2010.

[87] Hayashi, F., *Econometrics*, Princeton University Press, 2000, pp.7-12.

APPENDIX A

TUTORIAL ON DATA GENERATION CODE

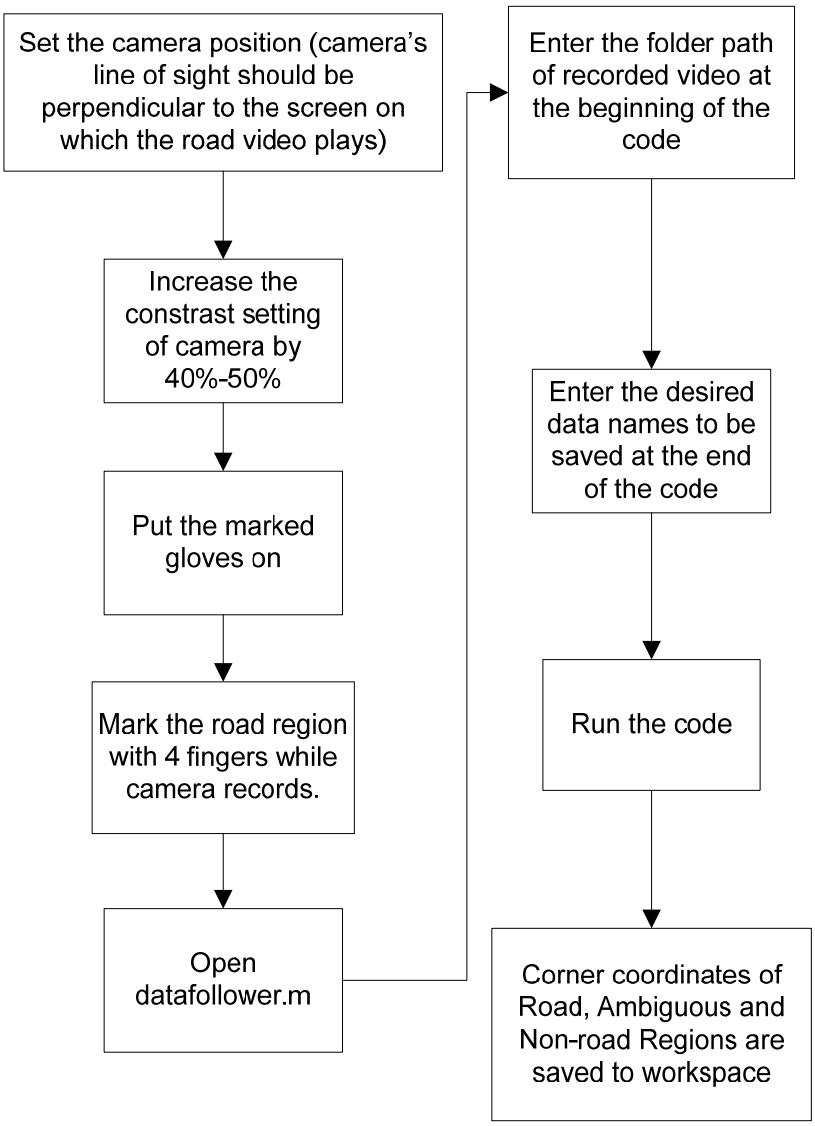

Figure A.1: Flow chart for ANN data generation code

APPENDIX B

TUTORIAL ON VANISHING POINT CODE

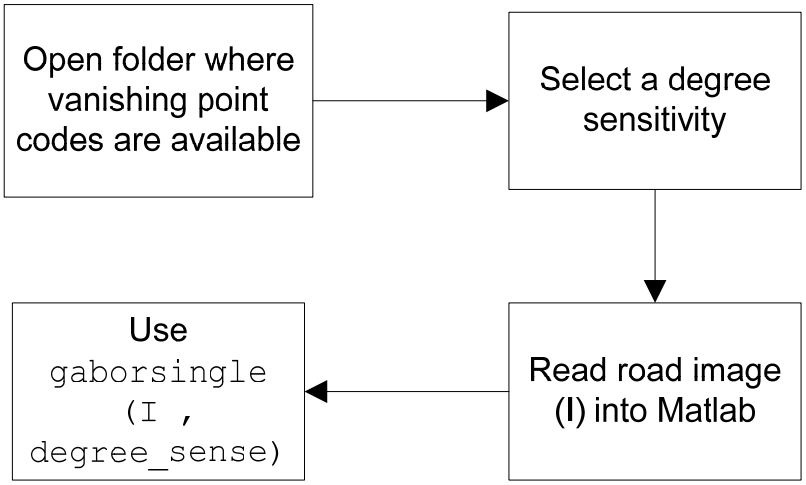

Figure B.1: Flow chart for dominant orientation detection

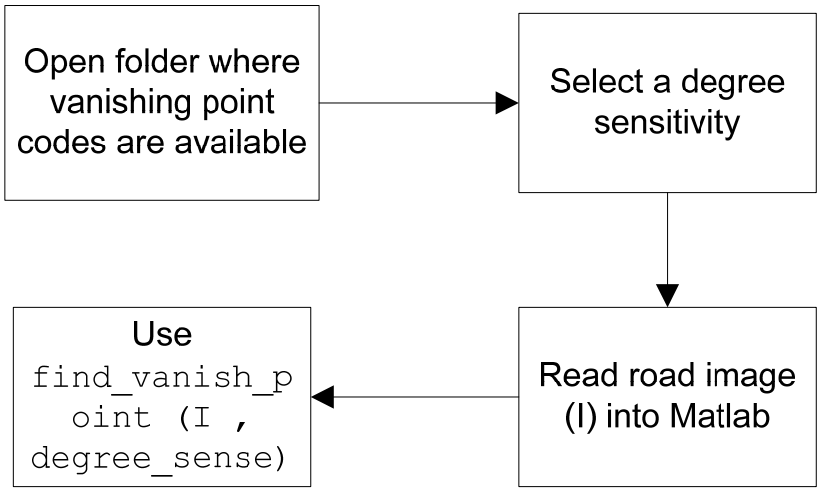

Figure B.2: Flow chart for vanishing point detection

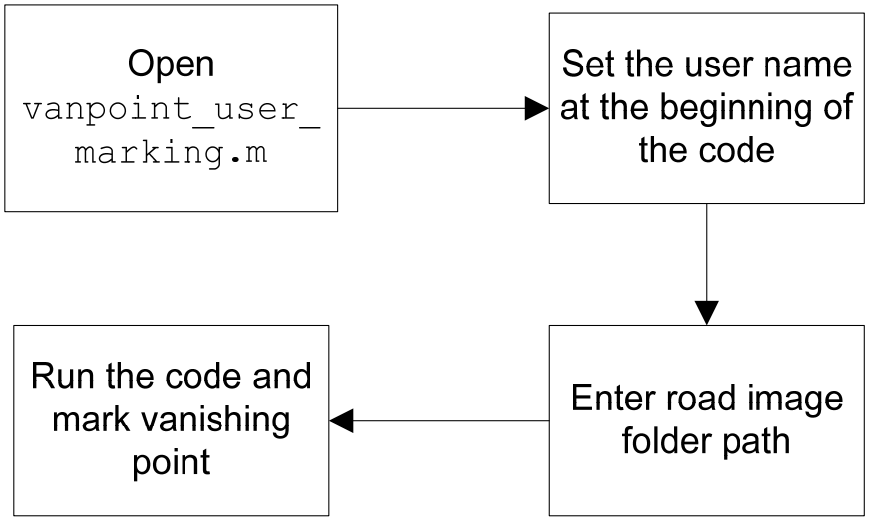

Figure B.3: Flow chart for vanishing point truth set generation

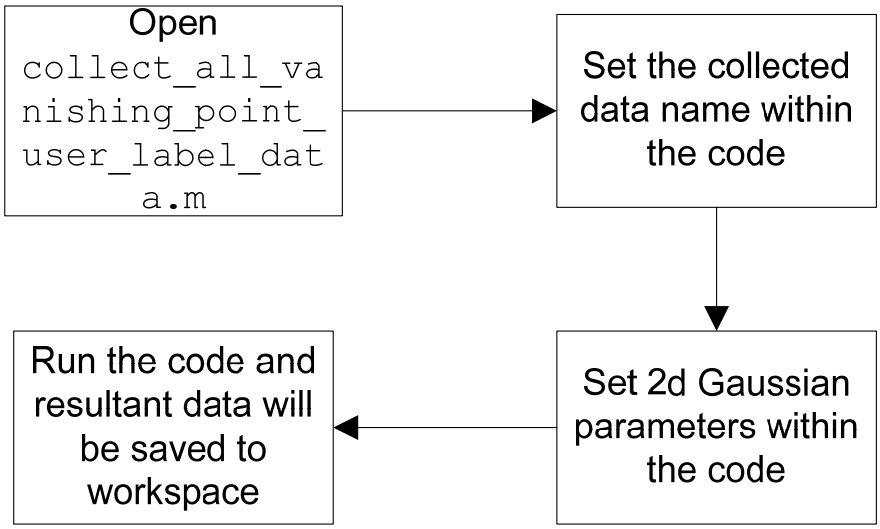

Figure B.4: Flow chart for evaluation of vanishing point truth sets

APPENDIX C

TUTORIAL ON COLOR BASED METHOD CODE

C.1 INPUT FEATURE VECTOR GENERATION

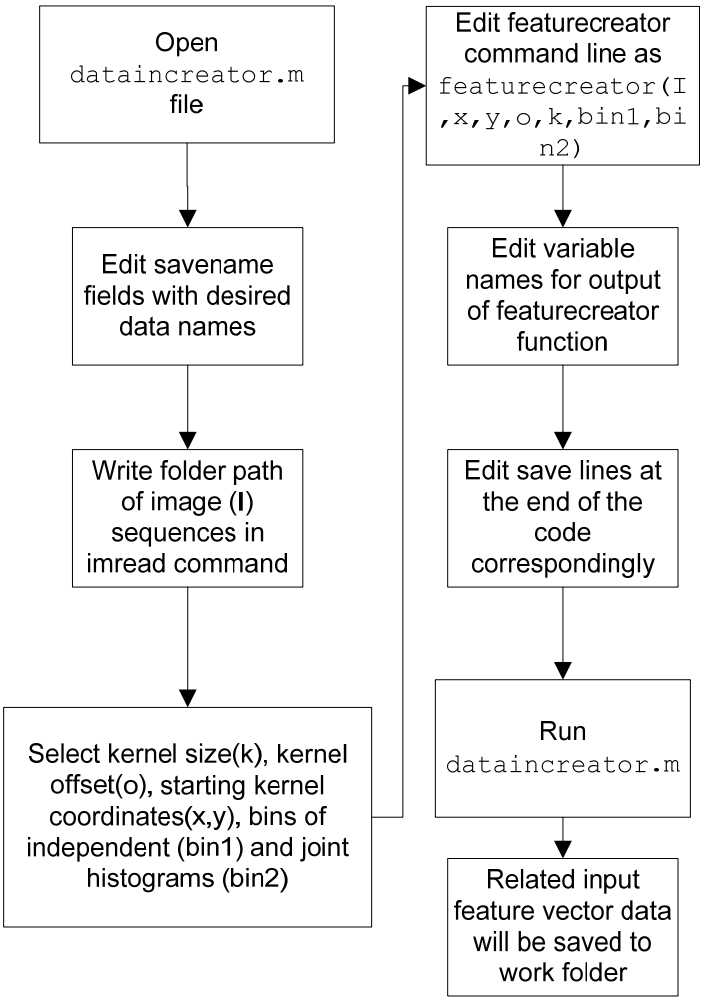

Figure C.1: Flow chart for input feature vector generation code

C.2 OUTPUT FEATURE VECTOR GENERATION

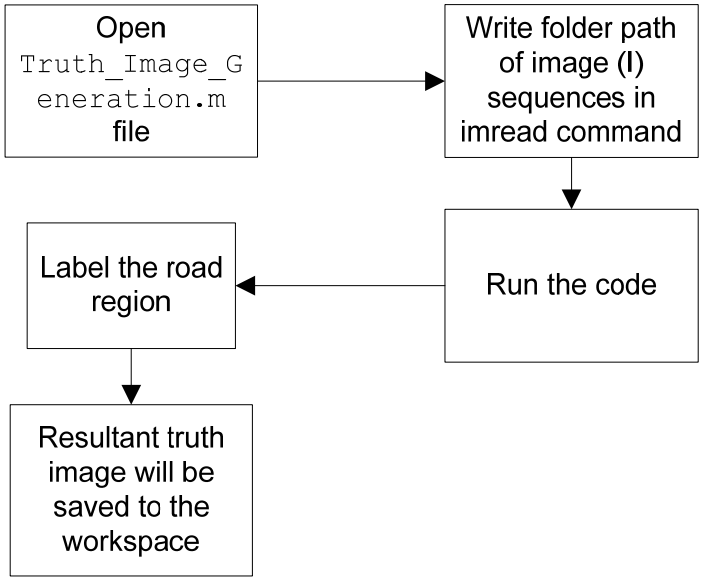

Figure C.2: Flow chart for truth image generation

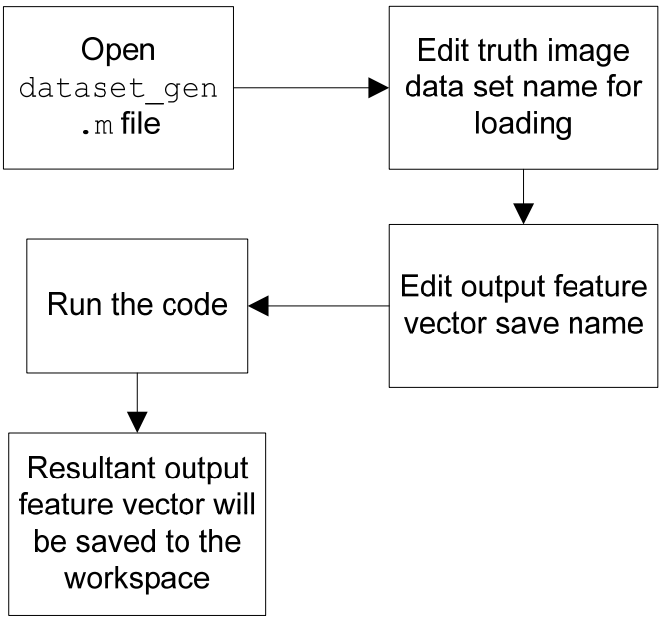

Figure C.3: Flow chart for output feature vector generation code

C.3 TRAINING

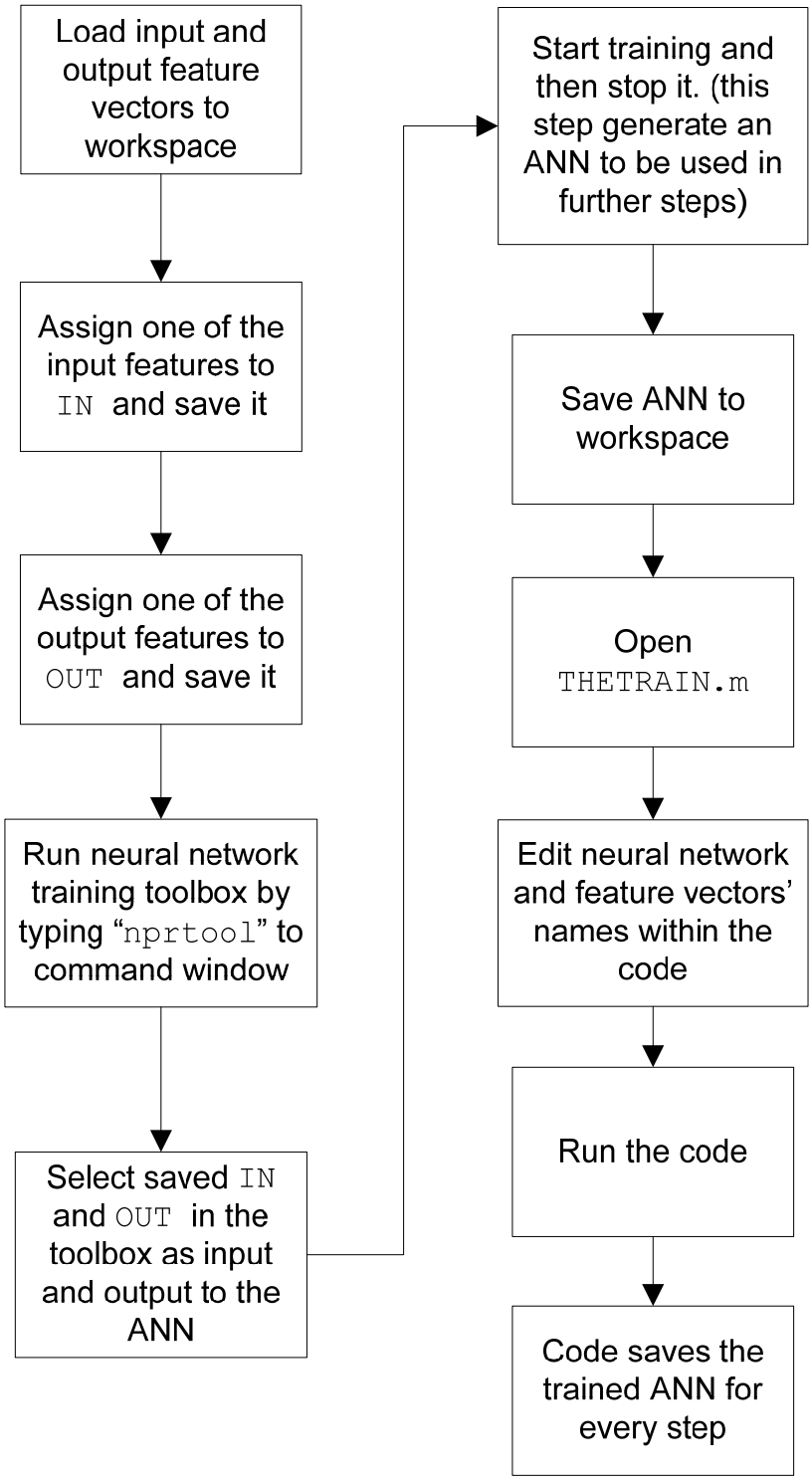

Figure C.4: Flow chart for training code

C.4 RESULT GENERATION

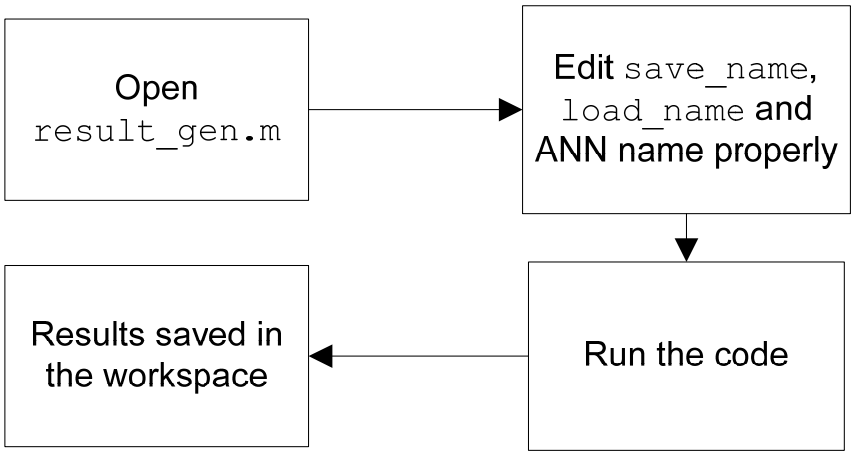

Figure C.5: Flow chart for result generation code

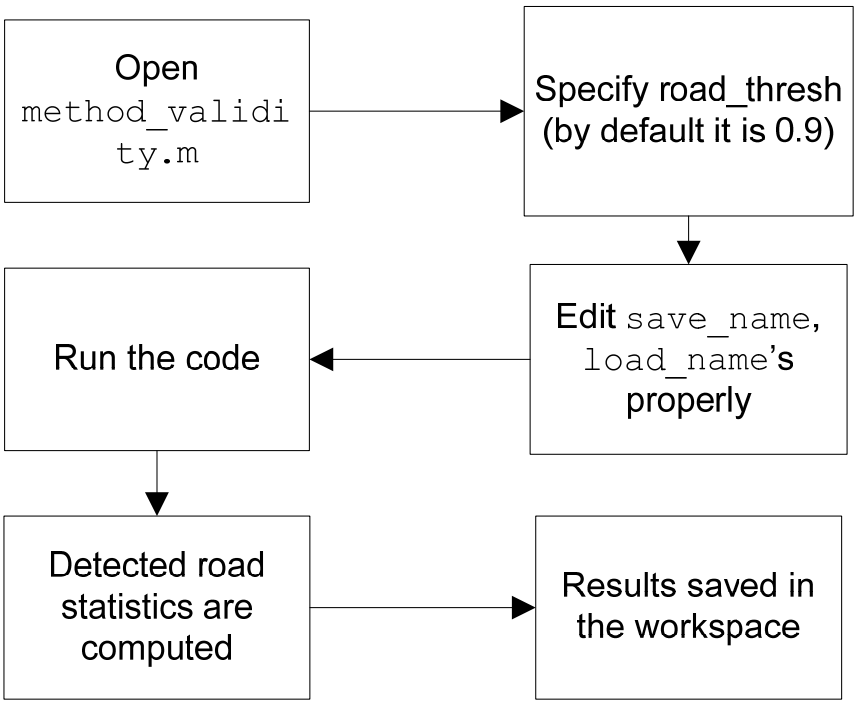

Figure C.6: Flow chart for method validity code

APPENDIX D

LIST OF M-FILES AND THEIR DESCRIPTIONS

collect_all_vanishing_point_user_label_data
Description: This m-file combines or user labeled vanishing point data and fits a 2d Gaussian curve on every related image and saves these images.

Inputs: vanishing point labelling data

Outputs: images with fitted 2D Gaussian curves

datafollower:
Description: This m file extracts the road, non-road and ambiguous regions from a road video which is labelled by hand. The m file is composed of 11 subsections. These are;

1) Read Movie File
   - Enter the path of the video here.
   - This part reads the hand-labelled movie and divides into frames

2) HSV Threshold Detection
   - Select the regions including finger tips.
   - This part calculates the thresholding values for extraction of fingertips.

3) Get Projection Information
   - User selects the corners of the screen that the video plays on.
   - User enters the corner coordinates of the image plane to where user wants to project the coordinates on the screen.

4) Find Homography Matrix for Projection
   - This part calculates the homography matrix in order to project the distorted screen coordinates into image coordinates [1 1 480 640].

5) Find Locations of Markers

- This part finds the location of fingertip markers using the threshold information calculated in part 2.

6) Cluster Marker Points into 4 Group

- Due to illumination problems program can find more than 4 fingertips.
- This part clusters this found markers into 4 points.

7) Project the Coordinates to 640x480

- Using the homography matrix, extracted points are projected into image coordinates.

8) Find Conservative Road and Non-Road Regions

- ROAD and NON-ROAD regions are extracted.
- bwidth=20 is default. and can be changed from this part.

9) Create Resultant Video Outside

- For visual purposes an output video can be created.

10) Clear Excess Variables

11) Save the Data

- Write here the desired data name to be saved.

Inputs: Path of the recorded labeling video

Outputs: Road – Non-road regions

dataincreator:

Description: This m-file creates input feature vectors of the specified RGB channel properties for neural network training using the image sequences saved in a directory with proper naming convention.

Inputs: path for the benchmark road images folder

Outputs: -

dataincreator_sat_Val:
Description: This m-file creates input feature vectors of the specified HSV channel properties for neural network training using the image sequences saved in a directory with proper naming convention.

Inputs: path for the benchmark road images folder

Outputs: -

dataset_gen:
Description: This m-file creates output feature vectors for neural network training using the truth matrices saved in a directory with proper naming convention.

Inputs: path for truth matrices folder

Outputs: vectorized output feature vectors

draw_line:
Description: This function fills a zeros' matrix with 1's as if it draws a line in the matrix starting from the desired position and toward the desired direction.

Inputs: desired orientation, desired position, size of the zeros matrix

Outputs: resultant line matrix

featurecreator:
Description: This function takes RGB image as input and creates input feature vectors according to the given inputs

Inputs:
I = RGB image
x_init = x-position (lefttop) of the first box
y_init = y-position (lefttop) of the first box
kernelsize = size of the square box that the feature vectors are extracted
offset = shift size between the boxes
bin1 = # of bins used for first feature set
bin2 = # of bins used for second feature set

Outputs:
feature1all = (bin1^3+2) x n size feature vector composed of the joint histogram of (bin1) bins + x-y coordinate of the box
feature2all = (3*bin2+2) x n size feature vector composed of the independent histogram of (bin1) bins + x-y coordinate of the box
featcombined = (bin1^3+3*bin2+2) x n size feature vector composed of the joint and independent histograms as in previous feature vectors + x-y coordinate of the box

featurecreator_sat:
Description: This function takes HSV image as input and creates input feature vectors according to the given inputs

Inputs:
I = HSV image
x_init = x-position (lefttop) of the first box
y_init = y-position (lefttop) of the first box
kernelsize = size of the square box that the feature vectors are extracted
offset = shift size between the boxes
bin1 = # of bins used for first feature set
bin2 = # of bins used for second feature set

Outputs:
feature1all = (bin1^3+2) x n size feature vector composed of the joint histogram of (bin1) bins + x-y coordinate of the box
feature2all = (3*bin2+2) x n size feature vector composed of the independent histogram of (bin1) bins + x-y coordinate of the box
featcombined = (bin1^3+3*bin2+2) x n size feature vector composed of the joint and independent histograms as in previous feature vectors + x-y coordinate of the box

find_H_matrix:
Description: This function finds the homography transformation matrix between coordinate systems specified by prjcoor (available coordinate system) and prjwanted (desired coordinate system).

Inputs: prjcoor (corner coordinates in distorted screen in the form of [u1 v1 ; u2 v2; u3 v3; u4 v4]), prjwanted (corner coordinates of the image plane in the form of [u1s v1s ; u2s v2s; u3s v3s; u4s v4s])

Outputs: H (Homography matrix)

find_HSV_matrix:
Description: This function finds a threshold for filtering out the RGB color channels of the fingertips in QOSS method.

Inputs: samp_im (an image matrix)

Outputs: themean (mean value of three channels for four fingertips), Thr (Threshold band value around the calculated mean)

find_vanish_point:
Description: This function finds the vanishing point coordinates of the input road image

Inputs: image, degree sensitivity

Outputs: coordinates of vanishing point

gaborfilt
Description: This function filters the image with previously built-up Gabor filter

Inputs: image, the orientation of the Gabor filter

Outputs: related Gabor filter, overall filtering result.

gaborsingle
Description: This function finding dominant texture directions of every pixel in an image

Inputs: image, degree sensitivity

Outputs: matrix of dominant direction orientation angles

gridded_set:
Description: This function converts a binary image into a sequence of intensity vectors

Inputs: I (truth matrix)

Outputs: sequence of intensity vectors

head_direction

Description: This function finds a heading vector

Inputs: output of trained neural network

Outputs: heading vector

jointhist:

Description: This function generates joint and independent histograms of given image in specified bins for RGB color space

Inputs:
I = RGB image
x_init = x-position (lefttop) of the first box
y_init = y-position (lefttop) of the first box
kernelsize = size of the square box that the feature vectors are extracted
offset = shift size between the boxes
feature1bin = # of bins used for joint histogram feature set
feature2bin = # of bins used for independent histogram feature set

Outputs:
feature1 = Joint histogram feature vector
feature2 = Independent histogram feature vector

jointhistsat:

Description: This function generates joint and independent histograms of given image in specified bins for HSV color space

Inputs:
I = HSV image
x_init = x-position (lefttop) of the first box
y_init = y-position (lefttop) of the first box
kernelsize = size of the square box that the feature vectors are extracted
offset = shift size between the boxes
feature1bin = # of bins used for joint histogram feature set
feature2bin = # of bins used for independent histogram feature set

Outputs:
feature1 = Joint histogram feature vector
feature2 = Independent histogram feature vector

method_validity:
Description: This m-file generates statistical information on the results of the trained neural network by comparing it with the truth matrices.

Inputs: trained neural network results, truth matrices

Outputs: following statistical information

project_coords:
Description: This function is used to project given coordinates according to the given homography transformation matrix.

Inputs: coords (coordinates to be projected), H (homography matrix to be used in projection)

Outputs: prj_coords (projected coordinates)
CORRECTLY_FOUND_rp = road pixels that are identified correctly (# of pixels which are found as road pixel when it is a road pixel in the truth image)
INCORRECTLY_FOUND_rp = road pixels that are identified incorrectly (# of pixels which are found as road pixel when it is a non-road pixel in the truth image)
CORRECTLY_FOUND_nrp = non-road pixels that are identified correctly (# of pixels which are found as non-road pixel when it is a non-road pixel in the truth image)
INCORRECTLY_FOUND_nrp = non-road pixels that are identified incorrectly (# of pixels which are found as road pixel when it is a non-road pixel in the truth image)
PCT_TRUE_POSITIVES = percentage of the true positives (percentage of correctly found road pixels)
PCT_TRUE_NEGATIVES = percentage of the true negatives (percentage of incorrectly found road pixels)
PCT_FALSE_POSITIVES = percentage of the true positives (percentage of correctly found road pixels)
PCT_FALSE_NEGATIVES = percentage of the true negatives (percentage of incorrectly found road pixels)

result_gen:
Description: This m-file generates a result matrix from the result vectors at the output end of neural network. Moreover generates figures visualizing the results and save them as .jpg's.

Inputs: zihni_sinir (trained neural network), FV (test input feature vectors)

Outputs: result (binary result image)

show_road_regions:
Description: This function is used to extract the road, non-road and ambiguous regions on the hand labelled video when the coordinates of quadleteral is given.

Inputs: cords_in (coordinates of corners of the quadliteral area), bwidth (width of the ambigious region band)

Outputs: ROAD (coordinates of corners of the quadliteral region enclosing the road), NONROAD1 (coordinates of corners of the quadliteral region enclosing the non-road region on the left side), NONROAD2 (coordinates of corners of the quadliteral region enclosing the non-road region on the right side), AMB1 (coordinates of corners of the quadliteral region enclosing the ambiguous region on the left side), AMB2 (coordinates of corners of the quadliteral region enclosing the ambiguous region on the right side)

THETRAIN:
Description: This function trains the previously generated neural network by continuously feeding the network with input and output feature vectors with the specified settings.

Inputs: neural network, input feature vectors, output feature vectors

Outputs: trained neural network

Truth_Matrix_Generation:
Description: This function creates truth matrices for road images whose directory path are specified.

Inputs: road image directory path

Outputs: truth matrix of the road image

vanpoint_test

Description: This routine finds all vanishing points for the entire image database.

Inputs: path of image database folder

Outputs: vanishing point result matrices

vanpoint_user_marking

Description: This m-file allows user to mark vanishing points for the entire image database. The results will be used for method evaluation.

Inputs: -

Outputs: labelled vanishing point data

voting

Description: This function votes the vanishing point candidates and output the vote matrix.

Inputs: dominant orientation matrix

Outputs: vote matrix

find_dir

Description: This function finds average texture orientation in an image

Inputs: dominant orientation matrix

Outputs: average orientation angle of the image